1994-06

Double Eulerian cycles on de Bruijn digraphs

Krahn, Gary William

Monterey, California. Naval Postgraduate School

http://hdl.handle.net/10945/42901

1

NAVAL POSTGRADUATE SCHOOL
Monterey, California

135pg

DISSERTATION

Double Eulerian Cycles on de Bruijn Digraphs

by

Gary William Krahn

June 1994

Dissertation Supervisor: Harold Fredricksen

Approved for public release; distribution unlimited

REPORT DOCUMENTATION PAGE

Form Approved OMB Np. 0704

Public reporting burden for this collection of information is estimated to average 1 hour per response, including the time for reviewing instruction, searching existing data sources, gathering and maintaining the data needed, and completing and reviewing the collection of information. Send comments regarding this burden estimate or any other aspect of this collection of information, including suggestions for reducing this burden, to Washingon headquarters Services, Directorate for Information Operations and Reports, 1215 Jefferson Davis Highway, Suite 1204, Arlington, VA 22202-4302, and to the Office of Management and Budget, Paperwork Reduction Project (0704-0188) Washington DC 20503.

1. AGENCY USE ONLY *(Leave blank)*

2. REPORT DATE
June 1994

3. REPORT TYPE AND DATES COVERED
Ph.D. Dissertation

4. TITLE AND SUBTITLE
DOUBLE EULERIAN CYCLES ON DE BRUIJN DIGRAPHS

5. FUNDING NUMBERS

6. AUTHOR(S) Gary William Krahn

7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES)
Naval Postgraduate School
Monterey CA 93943-5000

8. PERFORMING ORGANIZATION REPORT NUMBER

9. SPONSORING/MONITORING AGENCY NAME(S) AND ADDRESS(ES)

10. SPONSORING/MONITORING AGENCY REPORT NUMBER

11. SUPPLEMENTARY NOTES The views expressed in this thesis are those of the author and do not reflect the official policy or position of the Department of Defense or the U.S. Government.

12a. DISTRIBUTION/AVAILABILITY STATEMENT
Approved for public release; distribution is unlimited.

12b. DISTRIBUTION CODE

13.ABSTRACT *(maximum 200 words)*

A binary de Bruijn sequence has the property that every n-tuple is distinct on a given period of length 2^n. An efficient algorithm to generate a class of classical de Bruijn sequences is given based upon the distance between cycles within the Good - de Bruijn digraph. The de Bruijn property on binary sequences is shown to be a *randomness* property of the ZERO and ONE run sequences. Utilizing this randomness we find additional new structure in de Bruijn sequences. We analyze binary sequences that are not de Bruijn but instead possess the sufficient structure so that every distinct binary n-tuple can be systematically "combed" out of the sequence. These *complete* or *nonclassical de Bruijn* sequences are a generalization of the well-known de Bruijn cycle.

Our investigation focuses on binary sequences, called *double Eulerian cycles*, that define a cycle along a graph (digraph) visiting each edge (arc) exactly twice. A new algorithm to generate a class of double Eulerian cycles on graphs and digraphs is found. Double Eulerian cycles along the binary Good - de Bruijn digraph are partitioned by the run structure of their defining sequences. This partition allows for a statistical analysis to determine the relative size of the set of complete cycles defined by the sequences we study. A measure that categorizes double Eulerian cycles along graphs (digraphs) by the distance between the two visitations of each edge (arc) is provided. An algorithm to generate double Eulerian cycles of minimum measure is given.

14. SUBJECT TERMS de Bruijn Cycles, Eulerian Cycles, Shift Register, Binary Sequences, Digraphs, Graphs, Random Sequences, Good - de Bruijn Digraph

15. NUMBER OF PAGES 146

16. PRICE CODE

17. SECURITY CLASSIFICATION OF REPORT
Unclassified

18. SECURITY CLASSIFICATION OF THIS PAGE
Unclassified

19. SECURITY CLASSIFICATION OF ABSTRACT
Unclassified

20. LIMITATION OF ABSTRACT
UL

Approved for public release; distribution is unlimited.

Double Eulerian Cycles on de Bruijn Digraphs

by

Gary William Krahn
Lieutenant Colonel, United States Army
B.S., United States Military Academy, 1977
M.S., Naval Postgraduate School, 1985

Submitted in partial fulfillment of the
requirements for the degree of

DOCTOR OF PHILOSOPHY IN APPLIED MATHEMATICS

from the

NAVAL POSTGRADUATE SCHOOL
June 1994

Author: [redacted]

[redacted]
Craig W. Rasmussen
Assistant Professor of Mathematics

[redacted]
Siriphong Lawphongpanich
Associate Professor of Operations Research

[redacted]
Yutaka Kanayama
Professor of Computer Science

[redacted]
Richard Franke
Professor of Mathematics

[redacted]
Harold M. Fredricksen
Professor of Mathematics
Dissertation Supervisor

Approved by: [redacted]
Richard Franke, Chairman, Department of Mathematics

Approved by: [redacted]
Richard S. Elster, Dean of Instruction

Abstract

A binary de Bruijn sequence has the property that every n-tuple is distinct on a given period of length 2^n. An efficient algorithm to generate a class of classical de Bruijn sequences is given based upon the distance between cycles within the Good - de Bruijn digraph. The de Bruijn property on binary sequences is shown to be a *randomness* property of the ZERO and ONE run sequences. Utilizing this randomness we find additional new structure in de Bruijn sequences. We analyze binary sequences that are not de Bruijn but instead possess the sufficient structure so that every distinct binary n-tuple can be systematically "combed" out of the sequence. These *complete* or *nonclassical de Bruijn* sequences are a generalization of the well-known de Bruijn cycle.

Our investigation focuses on binary sequences, called *double Eulerian cycles*, that define a cycle along a graph (digraph) visiting each edge (arc) exactly twice. A new algorithm to generate a class of double Eulerian cycles on graphs and digraphs is found. Double Eulerian cycles along the binary Good - de Bruijn digraph are partitioned by the run structure of their defining sequences. This partition allows for a statistical analysis to determine the relative size of the set of complete cycles defined by the sequences we study. A measure that categorizes double Eulerian cycles along graphs (digraphs) by the distance between the two visitations of each edge (arc) is provided. An algorithm to generate double Eulerian cycles of minimum measure is given.

CONTENTS

LIST OF TABLES

LIST OF FIGURES

Table of Symbols and Notation

This section consists of a list of frequently used mathematical symbols and notation.

SETS	SYMBOL	MEANING
	$x \in S$	x is a member of S
	$x \notin S$	x is a not a member of S
	$S \subset T$	S is a proper subset of T
	$S \subseteq T$	S is a subset of T
	$\emptyset$	The empty set
	$A \cap B$	Intersection of sets A and B
	$A \cup B$	Union of sets A and B
	$A - B$	Difference of sets A and B (relative complement of B in A)
	$\|A\|$	Cardinality of A
	$V \times T$	Cartesian product of sets V and T
	$(s_1, \ldots, s_n)$	n-tuple
	$\{s_1, \ldots, s_n\}$	List of elements in a set
INTEGERS		
	$\mathbf{Z}^+$	Set of positive integers
	$a \| b$	a divides b
	$a \bmod b$	Remainder when a is divided by b
	$a \equiv b \bmod m$	a is congruent to b modulo m

FUNCTIONS

$\lfloor x \rfloor$	The greatest integer less than or equal to x
$\lceil x \rceil$	The least integer greater than or equal to x
ϕ	Euler's totient function
$\max(x, y)$	Maximum of x and y
$\min(x, y)$	Minimum of x and y

COUNTING

$P(n, r)$	Number of r-permutations of a set with n elements
$C(n, r)$	Number of r-combinations of a set with n elements
$C(n; n_1, n_2, \ldots, n_m)$	Multinomial coefficient $\binom{n}{n_1, n_2, \ldots, n_m} = \frac{n!}{n_1! n_2! \cdots n_m!}$
$\binom{n}{r}$	Binomial coefficient

GRAPHS AND DIGRAPHS

(x, y)	Directed edge or arc
xy	Undirected edge
$\|G\|_\rho$	Number of distinct walks in the graph G that traverse each edge exactly ρ times
$d_c(x, y)$	Distance from vertex x to vertex y on a cycle
$d_D(x, y)$	Directed distance from vertex x to vertex y
$d_e(x, e)$	Edge-distance from vertex x to edge e
$d(x, y)$	Distance from vertex x to vertex y
$d(x, V)$	Distance from vertex x to the set of vertices V where $d(x, V) = \min_{v \in V} d(x, v)$
$d_e(x, E)$	Edge-distance from vertex x to the set of edges E where $d_e(x, E) = \min_{e \in E} d_e(x, e)$

$M(G)$	Median of the graph G
$EM(G)$	Edge-median of the graph G
s_v	Status of vertex v
es_v	Edge-status of vertex v
σ-edges	Single-edges, i.e., edges that have been currently traversed once during a walk along a graph or digraph
out(x)	Outdegree of vertex x
in(x)	Indegree of vertex x
$\rho_v(F)$	Posture for a vertex v in the factor F
$M_F(D)$	Mean of the digraph D for the given factor F
B_n	The Good-de Bruijn digraph of order 2^n
$D_{(\lambda)}$	Graph D with each edge having multiplicity of λ
$\overline{B}_{n-(k+1)}$	Graph isomorphic to $B_{n-(k+1)}$
(α)	The cycle to which the edge α belongs
$(\alpha)_v$	Set of vertices on the cycle (α)
${}_y(\alpha)_v$	The set of vertices $v_i \in (\alpha)_v$ for which $d(y, v_i) = d_e(y, (\alpha))$
$(\alpha)_R$	The representative vertex on a cycle (α)
$\mathcal{W}_\gamma$	The set of walks that traverses each edge of a graph exactly γ times
$\mathcal{W}_v$	The set of walks beginning at vertex v
$\mathcal{W}_{2_v}$	The set of double Eulerian walks beginning at vertex v
$G = (V, E)$	Graph G with vertex set V and edge set E
$D = (V, A)$	Digraph D with vertex set V and arc set A

BOOLEAN ALGEGRA

B^n	Binary n-tuple
$\hat{\mathbf{x}}$	Conjugate of x
$\mathbf{x}'$	Conjugate of x
$\bar{\mathbf{x}}$	Complement of x
$\oplus$	Addition modulo 2

GENERAL

FSR_n	Feedback shift register of span n
PCR_n	Pure cycle register of span n
$\mathcal{S}_n$	The set of de Bruijn sequences of length 2^n
$\hat{\mathcal{S}}_n$	The set of balanced binary sequences of length 2^n
$\tilde{\mathcal{S}}_n$	The set of binary sequences of length 2^n with the run property
Run of ZEROs	A subsequence of consecutive 0's that is preceeded and followed by a 1.
Run of ONEs	A subsequence of consecutive 1's that is preceeded and followed by a 0.
$T(C,S)$	A sequence of n-tuples extracted from a 2^n-long sequence, S, by an (l,n)-comb, C
$r_{D_n}(k)$	The repetition number of the element k in the multiset D_n
log	Logarithm to base 2
TTIR	Abbreviation for: to the immediate right
:=	Assignment operator
$\underline{|||\bullet\bullet\bullet|}$	Depiction of a $(7,4)$-comb specified by $f_i(s_i, s_{i+1}, \ldots, s_{i+6}) = (s_i, s_{i+1}, s_{i+2}, s_{i+6})$

ACKNOWLEDGEMENTS

As in any multi-year project, there are many people to thank. I would like to thank the members of the staff and faculty in the Department of Mathematics at the Naval Postgraduate School who helped me often on a daily basis. Craig Rasmussen, Van Hensen, and David Canright introduced me to LaTeX and answered a myriad of technical questions. Thanks go to the members of my committee; they are Hal Fredricksen, Craig Rasmussen, Richard Franke, Siriphong Lawphongpanich, and Yutaka Kanayama. Each, in their own way, provided encouragement and guidance during all phases of the dissertation.

I would also like to acknowledge the support of Craig Rasmussen during the past three years. He taught me so much and provided many helpful suggestions throughout the preparation of this dissertation. Without his critical reading this documentation would be less than it now is, both in content and in style. His dedication served me well. More importantly, he became a good friend.

I am forever grateful to my advisor Professor Hal Fredricksen, for without him there would be no dissertation. He showed me the great beauty of mathematics, a vision I will keep for the rest of my life. Our friendship has continued for nearly ten years and has been an important part of my life. He always reminded me of the most

important issues in life, teaching me things not found in books. His confidence in this work always exceeded my own.

Finally, I want to acknowledge my wife and best friend, Paula. She cared for my well-being and has given me so much. Without her this endeavor would never have been accomplished.

I. PERSPECTIVE

"Begin at the beginning," the King said, very gravely, "and go on till you come to the end: then stop."

Alice's Adventures in Wonderland, Chapter 12

A. OVERVIEW

The advent of modern high-speed communication hardware creates a need for high-speed techniques to generate *random-like* sequences. Most digital computers and many communication systems handle information in binary form. One of the simplest and most efficient devices for generating deterministic, random looking binary sequences is the shift register. Every periodic binary sequence is obtainable from some suitably constructed shift register. This generality allows great versatility in shift register applications. The applications for shift register sequences include secure data transmission [Ref. 1], robot path planning [Ref. 2], multiple address coding [Ref. 3], error correcting codes [Ref. 4], radar range measuring [Ref. 3], and random number generation [Ref. 5].

It is well-known that a $(2^n - 1)$*-long sequence*, i.e., a binary sequence of length $2^n - 1$, containing all the non-zero binary n-tuples can be obtained from an n-span shift register by means of a feedback function consisting entirely of modulo 2 additions [Ref. 6]. Such functions are called *linear*. A great deal of theory has been developed about linear feedback shift registers. Finite fields provide the underlying mathematical foundation for the linear feedback shift register. Once we remove the restriction that the feedback function be linear, allowing products of variables, the fundamental structure of the shift rigister changes dramatically. As we expand from linear to nonlinear feedback functions the number of maximum length binary shift register sequences of degree n, 2^n-long sequences with distinct n-tuples, increases from less than $\frac{2^n}{n}$ to exactly $\frac{2^{2^{n-1}}}{2^n}$. Unlike in the linear case, for a general nonlinear feedback function it is often true that the best way to determine the resulting sequence is to exhaustively construct the state tree, since there are few algebraic approaches for the nonlinear analysis. Many fundamental questions remain to be answered for the nonlinear problem. There is a need for expanded techniques and tools.

In a sense, no finite length sequence is ever truly random. In particular, no sequence that depends on a rather small number of parameters, such as the feedback connections of a feedback shift register, can be considered truly random. These sequences, however, have the balance and run randomness properties expected of random sequences as defined by Golomb [Ref. 6]. We consider only sequences of finite length. Furthermore, the end of the sequence is considered to be contiguous with the beginning of the sequence, hence, *sequences* and *cycles* can be considered to be equivalent.

B. CURRENT RESEARCH

An important problem currently under consideration with respect to nonlinear sequences is that of finding a broad class of functions that yield near-maximal length cycles. Ford [Ref. 7], Lempel [Ref. 8], and Fredricksen [Ref. 9], among others, have provided methods for constructing de Bruijn cycles and others have expanded upon their ideas. Additionally, classes of functions that yield pairs of cycles of length 2^{n-1} are examined by Kibler [Ref. 10]. But here, as in so many other facets of the analysis of nonlinear functions, even though the structure suggests some underlying means of classifying these functions, the discriminating factor has proven to be elusive.

A *de Bruijn cycle* (or *de Bruijn sequence*) of length 2^n has the property that every n-tuple appears exactly once on a given period. For some applications it might not be necessary that the n bits of interest lie consecutively along the sequence. Our work examines an apparently unexplored area with respect to de Bruijn cycles. Although novel, this research has parallels to the current direction of study to find classes of de Bruijn cycles. We analyze sequences of length 2^n that are not de Bruijn but that have been found to possess sufficient structure so that every distinct binary n-tuple can be systematically "combed" out of the sequence - i.e., every n-tuple can be found appearing at the successive positions in a subsequence. We find that in this way the properties that make de Bruijn cycles so attractive can be extracted from sequences that *apparently* do not contain each n-tuple.

C. DESCRIPTION OF THE THESIS

A simple method to extract n-tuples from a sequence is to cycle the sequence through the pure cycling register of length 2^n (PCR_{2^n}) [Ref. 6], recording the bits of n arbitrary (not necessarily consecutive) but fixed registers at each shift of the PCR_{2^n}.

More precisely, we define a functional $\mathcal{F}(f_1, f_2, \ldots, f_{2^n})$, whose domain consists of the 2^n functions $(f_1, f_2, \ldots, f_{2^n})$. The function f_i projects the l bits $(s_i, s_{i+1}, \ldots, s_{i+l-1})$, $l \geq n$, from a 2^n-long binary sequence $(s_1, s_2, \ldots, s_{2^n})$ to a binary n-tuple. We restrict each of the functions f_i in $\mathcal{F}$ to be the projection from B^l to B^n defined by

$$f_i(s_i, s_{i+1}, \ldots, s_{i+l-1}) = (s_i, s_{j_1}, \ldots, s_{j_{n-2}}, s_{i+l-1}),$$

where $i < j_1 < \cdots < j_{n-2} < i + l - 1$.

It is useful to visualize each f_i as an l-long window containing n viewing stations or as a comb of length l with n surviving teeth, denoted as an (l, n)-*comb*. It should be noted that an (l, n)-comb is always to be applied to a sequence of length 2^n since our intent is to extract 2^n unique n-tuples.

The set of all 2^n-long *balanced* sequences, i.e., those with an equal number of 1's and 0's, is denoted by $\widehat{\mathcal{S}}_n$. If $S = 0001010001101111$, then S is an element of $\widehat{\mathcal{S}}_4$ that is clearly not de Bruijn, i.e., 4-tuples are repeated. Hence, the (4,4)-comb consisting of 4 consecutive teeth is not a comb for S. The (8,4)-comb specified by

$$f_i(s_i, s_{i+1}, s_{i+2}, s_{i+3}, s_{i+4}, s_{i+5}, s_{i+6}, s_{i+7}) = (s_i, s_{i+1}, s_{i+2}, s_{i+7}),$$

however, generates the following sequence of distinct 4-tuples from S: (0000, 0010, 0101, 1011, 0100, 1001, 0001, 0011, 0111, 1100, 1010, 0110, 1111, 1110, 1101, 1000).

It is convenient to depict combs by a symbol that portrays the teeth of the comb in a fairly obvious way. For example, the (8,4)-comb specified above by

$$f_i(s_i, s_{i+1}, s_{i+2}, s_{i+3}, s_{i+4}, s_{i+5}, s_{i+6}, s_{i+7}) = (s_i, s_{i+1}, s_{i+2}, s_{i+7})$$

is represented by the symbol $\underline{|||\bullet\bullet\bullet\bullet|}$. It can easily be verified that S has the following $(l,4)$-combs: $\underline{|||\bullet\bullet\bullet|}$, $\underline{||\bullet\bullet\bullet|\bullet|}$, $\underline{|\bullet|\bullet\bullet\bullet\bullet|\bullet\bullet\bullet|}$, $\underline{|\bullet|\bullet\bullet\bullet|\bullet\bullet\bullet\bullet|}$, $\underline{|\bullet|\bullet\bullet\bullet\bullet|\bullet\bullet\bullet\bullet|}$. (Note that these combs are distinct combs and are not cyclic shifts of each other.)

We see that a sequence can have several associated combs. Some sequences, however, have exactly one comb while other sequences have no combs at all. Clearly, a sequence of length 2^n composed of a de Bruijn cycle of length 2^{n-1} concatenated with itself cannot have a comb since, for each i,

$$f(s_i, s_{i+1}, \ldots, s_{i+l-1}) = f(s_{i+2^{n-1}}, s_{i+1+2^{n-1}}, \ldots, s_{i+l-1+2^{n-1}}).$$

For the same reason any sequence of length 2^n possessing a periodic part of period d, where d is a proper divisor of 2^n, cannot have a comb.

Fundamentally, the search for sequences and combs can proceed in either of two ways: either by finding a sequence satisfying a particular comb or by finding a comb that satisfies a particular sequence. In either case, the appropriate sequences are necessarily of length 2^n and are therefore generated by polynomials, over the field of two elements, of the form $f_k(x) = (x+1)^k$ for some $2^{n-1} < k \leq 2^n$ [Ref. 6]. Only those sequences from this set that are balanced need be considered.

Occasionally, we categorize an (l,n)-comb as an m-comb if at most m of its n teeth are consecutive. We establish a one-to-one correspondence between sequences and walks along a graph or digraph. We show that a sequence of length 2^n with an $(n-k)$ comb, $1 < k < n$, traverses each arc in the digraph $\overline{B}_{n-(k+1)}$ exactly 2^k times. (The digraph $\overline{B}_{n-(k+1)}$ is isomorphic to the Good - de Bruijn digraph $B_{n-(k+1)}$.) This property has far-reaching implica ›ns. We find that walks traversing each arc of the Good - de Bruijn digraph provide a common thread throughout each of the following chapters. The set of walks that visit each edge (arc) in a graph (digraph) exactly γ times is denoted by $\mathcal{W}_\gamma$.

The primary goal of the research presented here is to provide a better understanding of the de Bruijn property of distinct n-tuples, binary de Bruijn cycles, and the Good - de Bruijn digraph. The thesis consists of 7 chapters:

Chapter II: The purpose of this chapter is to make the thesis self-contained. We provide a concise introduction to graph theory, feedback shift registers, and their resulting sequences. This introduction is suited for the analysis of double Eulerian cycles, binary de Bruijn cycles, and the Good - de Bruijn digraph. Apart from basic definitions, Chapter II also introduces definitions that are not in common use but proved to be convenient in the nonlinear theory of periodic sequences. We introduce some initial analysis on ***double Eulerian cycles***, cycles that traverse each edge (arc) along a graph (digraph) exactly twice.

Chapter III: We investigate sequences called ***complete cycles*** or ***nonclassical de Bruijn cycles.*** A complete cycle of length 2^n has the property that each of the possible 2^n binary n-tuples lies along a fixed pattern or (l, n)-comb of the sequence. The analysis of these complete cycles is primarily concerned with combs where $n-1$ of the bits of interest lie consecutively along the sequence. A characterization of a class of complete cycles is made in terms of the walk they define along an appropriate Good - de Bruijn digraph. A statistical analysis is made to determine the number of complete cycles defined by a particular class of sequences.

Chapter IV: To analyze $(n-1)$-combs, i.e., those with $n-1$ consecutive teeth, some essential theory on double Eulerian cycles along graphs and digraphs is developed. We define a *measure* that categorizes double Eulerian cycles by a function of the two *visitations* along each edge in the traversed graph. In essence, the measure describes where particular n-tuples are located on the sequence in some rough sense. This measure parallels the discrete logarithm problem of finite fields [Ref. 11] that forms the foundation of some current public key cryptography systems. A conjecture is given for the minimum measure (or *value*) of a Good - de Bruijn digraph.

Chapter V: We describe how the de Bruijn property of distinct n-tuples results from a randomness property of the run lengths in a binary sequence. The run structure of the sequences defining an Eulerian or double Eulerian cycle along the Good - de Bruijn digraph is completely determined. A statistical analysis shows that the property of containing distinct n-tuples in a binary de Bruijn cycle of length 2^n is equivalent to the *Expected Value Property* for run lengths in a *random* binary sequence.

Chapter VI: The concepts derived in Chapter IV are used to develop a new algorithm to generate classical de Bruijn cycles. The algorithm is based on the edge-factors of the Good - de Bruijn digraph and the distance between cycles. The edge-factors of a graph are found to be an important element to determine the minimum measure of a graph or digraph.

Chapter VII: Here we summarize the results of the thesis. Directions for further research are identified and open problems are discussed.

In summary, the ensuing chapters emerge from the development of *combing* sequences in Chapter III. More importantly, each result is inspired by the desire to gain a greater understanding of nonlinear binary sequences in general.

D. HISTORICAL NOTE

PROBLEM: ***Given m symbols (which, without ιoss of generality, we take to be 0, 1, . . . , m-1) and a positive integer n, find a sequence of these symbols having minimum length, that when arranged as a cycle, contains every sequence of n consecutive symbols.***

A solution to this problem is a de Bruijn cycle of length $(m)^n$. In 1951 van Aardenne-Ehrenfest and de Bruijn [Ref. 12] showed that de Bruijn cycles exist for all $m \geq 2$ and $n \geq 1$. In fact the de Bruijn cycle problem has been independently rediscovered many times.

The problem of finding de Bruijn cycles for $m = 2$ became well-known through de Bruijn's paper [Ref. 13], where the number of solutions was found to be $2^{2^{n-1}-n}$. de Bruijn [Ref. 14], however, credits Stanley for discovering that the problem had been proposed and solved half a century earlier in the French problem journal *l'Intermédiaire des Mathématiciens* in 1894. The problem was proposed by de Rivière in 1894 and solved by Flye Sainte-Marie [Ref. 15] that same year. Flye Sainte-Marie found the same number, $2^{2^{n-1}-n}$, and his method of solution was paralleled by de Bruijn. Three years later, Mantel [Ref. 16] found a solution whenever m is prime using an algebraic method. After 1897, the problem was apparently entirely forgotten until 1934, when it was reintroduced by Martin [Ref. 17]. Martin approached the problem combinatorially and proved the existence of de Bruijn cycles for all m and n by creating an algorithm to construct such cycles. A decade after Martin, de Bruijn [Ref. 13] and Good [Ref. 18] independently rediscovered and solved the problem for the case $m = 2$ using graph theoretic and group theoretic concepts.

The corresponding problem for $m > 2$ symbols was first raised and solved in 1951 [Ref. 12]. The number of solutions was found to be $(m!)^{(m)^{n-1}}(m)^{-n}$ using methods of determinants on the adjacency matrix representing the Good - de Bruijn digraph.

Subsequently, algorithms to generate some or all of the de Bruijn cycles of length 2^n have been repeatedly uncovered. It is a palatial problem that will undoubtedly lure attempts to unravel it again and again.

II. DEFINITIONS AND NOTATION

> *"...But do cats eat bats, I wonder?" And here Alice began to get rather sleepy, and went on saying to herself, in a dreamy sort of way, "Do cats eat bats? Do cats eat bats?" and sometimes "Do bats eat cats?" for you see, as she couldn't answer either question, it didn't much matter which way she put it.*
>
> Alice's adventures in Wonderland, Chapter 1

A. INTRODUCTION

Before describing double Eulerian cycles, de Bruijn cycles, and the methods and techniques we need to address in our work, it is necessary to provide applicable definitions and notation. A detailed description of linear and nonlinear shift register sequences appears in Golomb [Ref. 6]. A text on graph theory such as Bondy and Murty [Ref. 19], provides a thorough discussion on the relevant material on graph theory. The reader who is thoroughly familiar with the vocabulary and concepts of graph theory and de Bruijn cycles may wish to skip most of this chapter.

B. GRAPHS

A *graph* G consists of a set $V = \{v_1, v_2, \ldots, v_p\}$, of elements called vertices (or nodes) and a set $E = \{e_1, e_2, \ldots, e_q\}$ of unordered pairs of vertices called *edges*. The edge between vertex x and vertex y is written as xy or yx. The graph G is said to have *order* p and *size* q. We write $G = (V, E)$ and say V is the *vertex set* and E is the *edge set*.

Edges of the form $e = v_iv_i$ are called *loops*. A graph without loops is called *simple* if it also has no pair of vertices forming more than one edge. A graph is *finite* if its order is finite. We will consider only graphs that are finite, but not necessarily simple.

Edge $e = v_iv_j$ is said to be *incident* to the vertices v_i and v_j. Similarly, v_i and v_j are incident to the edge e. Two vertices that are incident to a common edge are *adjacent*, as are two edges that are incident to a common vertex. A set of vertices (edges) is *independent* if the vertices (edges) are mutually nonadjacent. The graph $G' = G - \{v\}$ is the graph G with the vertex v and its incident edges removed.

A *subgraph* of a graph $G = (V, E)$ is a graph $H = (V', E')$ where $V' \subseteq V$ and $E' \subseteq E$. Suppose that V' is a nonempty subset of V. The subgraph of G whose vertex set is V' and whose edge set is all of those edges of G for which both incident vertices are in V' is called the subgraph of G *induced* by V'. Suppose that E' is a nonempty subset of E. The subgraph of G whose vertex set is the set of all vertices incident to the edges of E' and whose edge set is E' is called the subgraph of G *induced* by E'.

A *walk* $W = (v_1, e_1, v_2, e_2, \ldots, e_{k-1}, v_k)$ in G is an alternating sequence of vertices and incident edges, beginning and ending on a vertex of G. A walk may have repeated edges and repeated vertices. The *length* of a walk is the number of edges in the sequence. A walk can also be identified by merely listing the sequence of adjacent edges or, in a graph without multiple edges, by listing the sequence of adjacent vertices.

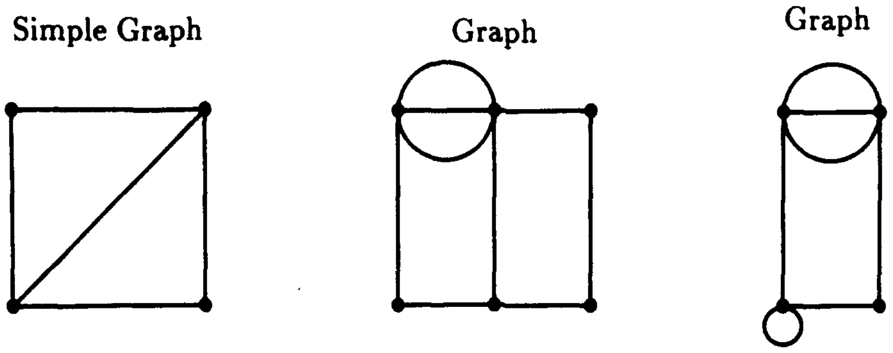


Figure 1. *Graphs and a simple graph*

A walk is said to be *closed* if the initial and terminal vertices are the same. A closed walk is called a *cycle.* When it is not important to designate the initial vertex of a cycle, the terminal vertex will typically not be listed in the sequence of adjacent vertices. Using this notation, any cyclic shift of a cycle is considered to be the same cycle. Further, this allows a distance to be defined between the vertices (edges) of a simple cycle. The distance between the vertices v_i and v_j on a simple cycle $C = (v_0, v_1, \ldots, v_{p-1})$ is denoted

$$d_c(v_i, v_j) = \min\left((i-j) \bmod p, (j-i) \bmod p\right).$$

Similarly, the distance between the edges e_i and e_j on a simple cycle $C = (e_0, e_1, \ldots, e_{p-1})$ is

$$d_c(e_i, e_j) = \min\left((i-j) \bmod p, (j-i) \bmod p\right).$$

A walk with distinct edges is called a *trail.* If the vertices of a trail are distinct, the walk is called a *path.* A closed path is called a *simple cycle.* If

$$(v_1, e_1, v_2, e_2, \ldots, e_{k-1}, v_k)$$

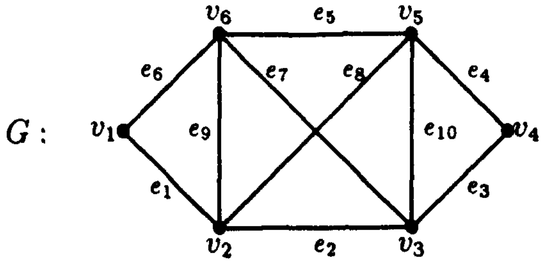


Figure 2. *Cycles*

is a walk, v_1 is the initial vertex, the set of *interior* vertices is $\{v_2, v_3, \ldots, v_{k-2}, v_{k-1}\}$, and v_k is the terminal vertex of the walk. That is, all of the vertices of the walk that occur somewhere other than as the initial and terminal vertices of the walk are interior. The set of all walks along a graph beginning at vertex v is denoted by $\mathcal{W}_v$. The set of all walks in which each of the edges appears exactly γ times is denoted by $\mathcal{W}_\gamma$. Additionally, the set of all walks in $\mathcal{W}_v$ where each of the edges appears exactly twice is denoted by $\mathcal{W}_{2_v}$.

A cycle C of *period* p contains p edges and we write $|C| = p$. If C is a cycle, but not a simple cycle, then C is said to be *reducible*. Evidently, a cycle is reducible if it is not a closed path. The closed path $(v_1, v_2, v_3, v_4, v_5, v_6)$ in Figure 2 is both a cycle and a simple cycle. The closed trail $C = (v_2, v_6, v_3, v_5, v_6, v_1)$ is a cycle but not a simple cycle, i.e., C is reducible. The cycle C can be reduced to the two simple cycles (v_2, v_6, v_1) and (v_6, v_3, v_5).

The *degree* of a vertex v in a graph, denoted $\deg(v)$, is the number of edges incident to v. A loop at a vertex contributes two to the degree of that vertex.

A graph G is said to be *connected* if, for all i, j, there exists a path connecting v_i and v_j. A graph that is not connected is said to be *disconnected.*

A trail in a graph G is called *Eulerian* provided it contains every edge of G exactly once. An *Eulerian cycle* in G is a cycle that includes each edge of G exactly once. A walk in a graph G is called *double Eulerian* if it contains every edge in G exactly twice. A path in a graph G is called *Hamiltonian* provided that it contains every vertex of G exactly once. A *Hamiltonian cycle* in G is a cycle that includes each vertex of G exactly once.

The following theorem is a well-known result in graph theory. See, for example, Bondy and Murty [Ref. 19].

Theorem II.1 *A connected graph G has a closed Eulerian trail if and only if the degree of each vertex is even.*

A *partition* of a set X is a family $\{X_i | i \in I \subseteq \mathbf{Z}^+\}$ of non-empty subsets of X such that $X = \bigcup_{i \in I} X_i$ and the family is pairwise disjoint.

A *factor* of a connected graph $G = (V, E)$ is a set of cycles in G that induces a partition of V. In a similar manner, an *edge-factor* of a connected graph G is a set of cycles in G that induces a partition of E. The set of cycles $\{(v_1, v_2, v_6), (v_3, v_4, v_5)\}$ is a factor of the graph G in Figure 2. The set of cycles $\{(v_1, v_2, v_6), (v_3, v_4, v_5), (v_2, v_5, v_6, v_3)\}$ is an edge-factor of G.

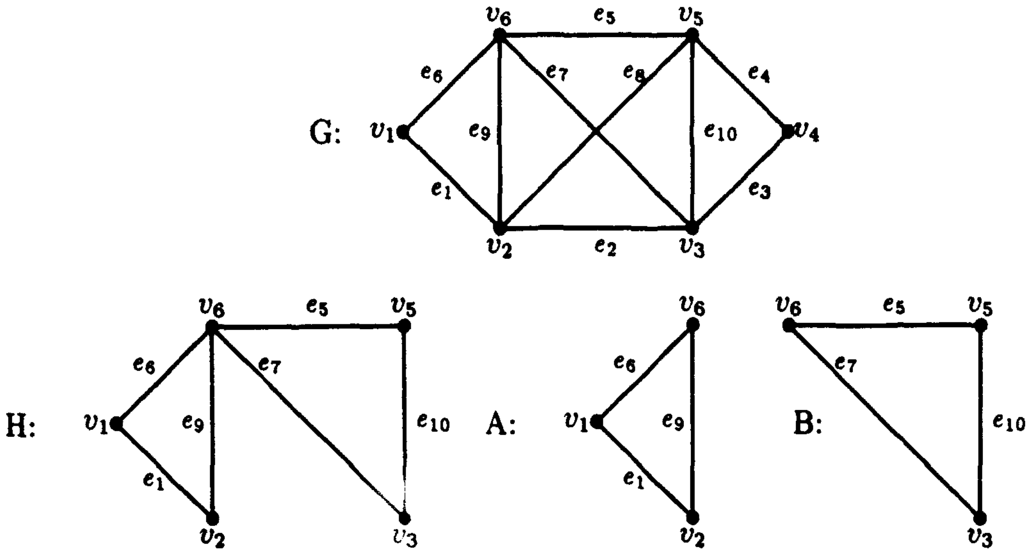


Figure 3. *A graph with a reducible cycle*

The subgraph induced by the edges of a reducible cycle has a nontrivial edge-factor. For example, let the edge-factor F of G in Figure 3 consist of the cycles $C_1 = (v_2, v_6, v_3, v_5, v_6, v_1)$ and $C_2 = (v_2, v_3, v_4, v_5)$. The subgraph H of G in Figure 3 is induced by the edges of C_1. Since C_1 is reducible, H has a nontrivial edge-factor consisting of the cycles $A = (v_2, v_6, v_1)$ and $B = (v_6, v_3, v_5)$. The set of cycles $\{C_2, A, B\}$ is an edge-factor of G, denoted by $F_{A,B}$.

A *tree* is a connected acyclic graph, i.e., the path connecting any two vertices is unique. Any vertex in a tree can be distinguished as the *root* of the tree. Every path from the root has a last or terminal vertex. The *height* of a rooted tree is the length of the longest path from the root.

Two graphs $G_1 = (V_1, E_1)$ and $G_2 = (V_2, E_2)$ are *isomorphic* if there is a bijection f from V_1 to V_2 such that $f(x)f(y)$ is an edge of G_2 if and only if xy is an edge of G_1.

The *distance*, $d(x, y)$, between two vertices x and y in a connected graph G is the minimum length of a path joining them. The *edge-distance*, $d_e(v, e)$, between a vertex v and an edge $e = xy$ in G is

$$d_e(v, e) = \min\left(d(v, x), d(v, y)\right).$$

Given a subset $X \subseteq V$ of vertices in $G = (V, E)$, the distance between a vertex $v \in V$ and X is

$$d(v, X) = \min_{x \in X} d(v, x). \qquad \text{(II.1)}$$

Similarly, given a subset $Y \subseteq E$ of edges, the edge-distance from v to Y is

$$d_e(v, Y) = \min_{y \in Y} d_e(v, y). \qquad \text{(II.2)}$$

Let $G = (V, E)$ be a connected graph. The *status*, s_v, of a vertex v in G is defined by $s_v = \sum_{n \in V} d(v, n)$. The *median* $M(G)$ of a graph G is the set of vertices of minimum status. In a similar manner, the *edge-status*, es_v, of a vertex $v \in G$ is defined as $es_v = \sum_{e \in E} d_e(v, e)$. The *edge-median* $EM(G)$ of a graph G is the set of vertices of minimum edge-status.

For example, in Figure 4 for the graph G we find $s_{v_1} = 9$, $s_{v_2} = 8$, $es_{v_1} = 4$, $es_{v_2} = 4$, $M(G) = \{v_2\}$, and $EM(G) = \{v_1, v_2\}$.

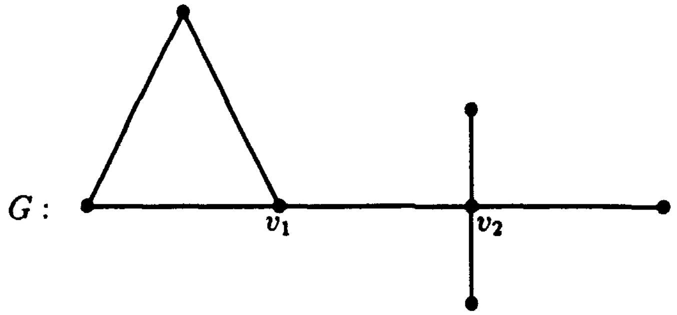


Figure 4. *A graph for which $M(G) \neq EM(G)$*

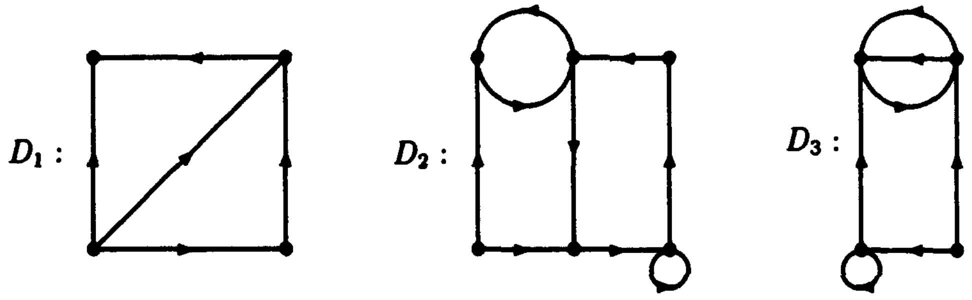


Figure 5. *Digraphs*

C. DIGRAPHS

A *digraph* (or *directed graph*) $D = (V, A)$ has a set V of elements called *vertices* and a set $A \subseteq V \times V$ of *ordered* pairs of (not necessarily distinct) vertices called *arcs*. We think of the arc $\alpha = (x, y)$ as *leaving* x and *entering* y, that is, directed from x to y; x is the *initial vertex* of α and y its *terminal vertex*. A digraph may contain each of the arcs (x, y) and (y, x) as well as loops of the form (x, x). A loop (x, x) enters and exits the same vertex x. A general digraph may also include multiple arcs.

A vertex of the digraph D has two associated degrees. The *outdegree* of a vertex v, denoted $\text{out}(v)$, is the number of arcs for which v is the initial vertex. The *indegree* of v, denoted $\text{in}(v)$, is the number of arcs for which v is the terminal vertex. The loop (x, x) contributes 1 to each of $\text{in}(x)$ and $\text{out}(x)$.

The definitions of walk, path, trail, factor, and cycle carry over from graphs to digraphs in a fairly obvious way. For example, a ***directed walk*** in a digraph $D = (V, A)$ is a sequence of vertices and arcs, $(v_1, \alpha_1, v_2, \ldots, \alpha_{k-1}, v_k)$, with the property that $(v_i, v_{i+1}) \in A$ for $1 \leq i \leq k-1$. A directed walk is a ***directed path*** if all of its vertices are distinct, a ***directed trail*** if all its arcs are distinct, and a ***closed directed walk*** if the initial and terminal vertices are the same. A closed directed walk is a *cycle*; a closed directed path is a *simple cycle*.

For any graph $G = (V, E)$ we obtain a digraph $D = (V, A)$ by giving each edge $xy \in E$ an orientation, that is, by replacing xy with either (x, y) or (y, x). Such a digraph D is called an *orientation* of G. A graph has $2^{|E|}$ orientations if it has no loops and $2^{|E|-k}$ orientations if G has k loops. Conversely, given a digraph $D = (V, A)$ we can remove the direction of its arcs thereby obtaining a graph $G = (V, E)$. Such a graph is called the *underlying graph of* G. A digraph D has exactly one underlying graph, denoted by G_D.

A *weak* digraph is one that has a connected underlying graph. A digraph $D = (V, A)$ is said to be *strong* provided that for each pair of distinct vertices $x, y \in V$ there is a directed walk from x to y and a directed walk from y to x (i.e., there is a

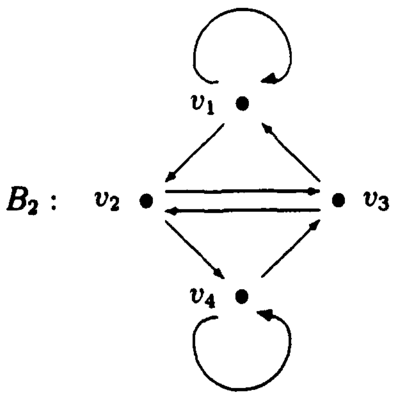


Figure 6. *An illustration of an edge-factor of a directed graph*

directed cycle that includes y and x). A directed trail in a digraph $D = (V, A)$ is called *Eulerian* provided that it contains every arc of A exactly once. A directed path is said to be *Hamiltonian* if it contains every vertex of V exactly once. A *Hamiltonian cycle* is one that contains every vertex of V exactly once, where the initial and terminal vertices may be considered the same.

A *factor* of a weak digraph $D = (V, A)$ is a set of cycles in D that induce a partition of V. Similarly, an *edge-factor* of a weak digraph D is a set of cycles in D that induce a partition of A.

As an example, the set of cycles $\{(v_1), (v_2, v_3), (v_4)\}$ in Figure 6 is a factor of graph B_2. The set of cycles $F = \{(v_1), (v_1, v_2, v_3), (v_2, v_4, v_3), (v_4)\}$ constitute an edge-factor of B_2.

For vertices x and y in a strong digraph D, the *directed distance* $d_D(x, y)$ is the length of a shortest directed path from x to y in D. The directed distance from any vertex to itself is zero, i.e., $d_D(x, x) = 0$. Unless the digraph D is symmetric, it is not generally the case that $d_D(x,y) = d_D(y,x)$ for all vertices x, y in V. Therefore, directed distance is not in general a metric. The *distance*, $d(x,y)$, between two vertices x and y in a weak directed D digraph is the minimum length of a path joining them in the *underlying graph* of D.

Given an edge-factor F of a digraph $D = (V, A)$, the *posture*, $\rho_v(F)$, of a vertex $v \in V$ is defined by

$$\rho_v(F) = \sum_{C \in F} 2|C| d_e(v, C), \tag{II.3}$$

where $|C|$ is the number of edges in the cycle C, $d_e(v,C) = \min_{x \in C} d_e(v,x)$ in G_D, the underlying graph of D, and the summation is over all of the cycles of the edge-factor.

The *mean* for the factor F of a directed graph D, denoted by $M_F(D)$, is defined to be the set of vertices with minimum posture for the edge-factor F. As an example, for the edge-factor F of B_2 we find $\rho_{v_1}(F) = 10$, $\rho_{v_2}(F) = 4$, and $M_F(B_2) = \{v_2, v_3\}$.

D. GOOD - DE BRUIJN GRAPHS AND DE BRUIJN CYCLES

The original formulation of the de Bruijn cycle problem for $m = 2$ can be viewed as finding the number of Hamiltonian paths in an appropriate directed graph. This graphical interpretation is very useful in understanding the properties of de Bruijn cycles.

A binary feedback shift register of span n (FSR$_n$) is a collection of n storage devices $(x_0, x_1, \ldots, x_{n-1})$, each capable of holding an element of $B = \{0, 1\}$, together with a feedback function $f(x_0, x_1, \ldots, x_{n-1})$, taking on a value of 0 or 1, computed from the contents of the n storage devices. The contents of the register at time t, regarded as a binary n-tuple or a binary vector, are called the *state* of the register. A FSR$_n$ has 2^n possible states, namely the elements of the set B^n of all binary n-tuples. The feedback function $f(\mathbf{x})$ of the FSR$_n$, where $\mathbf{x} = (x_0, x_1, \ldots, x_{n-1})$, induces a mapping F: $B^n \rightarrow B^n$, where $F(\mathbf{x}) = \mathbf{y}$ if and only if

$$y_i = \begin{cases} x_{i+1}; & i = 0, \ldots, n-2 \\ f(x); & i = n-1. \end{cases}$$

At the beginning of each time interval, determined by an external clock, there is a transition from one state to the next.

The superposition of all possible state transition graphs for each positive integer n defines the binary *Good - de Bruijn digraph* of order n, denoted by B_n. See Figure 7 for some examples of de Bruijn graphs. Thus, the Good - de Bruijn digraph B_n is a directed graph with 2^n vertices, each labeled with a unique binary vector of length n, and an arc $(\mathbf{x}, \mathbf{y})$ from vertex $\mathbf{x} = (x_0, x_1, \ldots, x_{n-1})$ to vertex $\mathbf{y} = (y_0, y_1, \ldots, y_{n-1})$ if and only if $(x_1, x_2, \ldots, x_{n-1}) = (y_0, y_1, \ldots, y_{n-2})$. We call $\mathbf{y}$ a *successor* of $\mathbf{x}$ and $\mathbf{x}$ a *predecessor* of $\mathbf{y}$. If $(\mathbf{x}, \mathbf{y}) \in A$, $\mathbf{x}$ is adjacent to $\mathbf{y}$ in B_n. The *conjugate* $\hat{\mathbf{x}}$ of the n-tuple $\mathbf{x} = (x_0, x_1, \ldots, x_{n-1})$ is defined as $\hat{\mathbf{x}} = (\overline{x}_0, x_1, \ldots, x_{n-1})$, where $\overline{x}_i = x_i \oplus 1$ and $\oplus$ denotes addition modulo 2. The *companion* $\mathbf{x}'$ of the n-tuple $\mathbf{x}$ is defined as $\mathbf{x}' = (x_0, x_1, \ldots, \overline{x}_{n-1})$.

Each arc in B_n can be viewed as an element of the $(n+1)$-dimensional binary vector space B^{n+1}. The arc between the vertex $(x_0, \ldots, x_{n-1})$ and vertex $(x_1, \ldots, x_n)$

in B_n can be labeled by the $(n+1)$-tuple $(x_0, x_1, \ldots, x_{n-1}, x_n)$. The total number of arcs in B_n is 2^{n+1}, and each $(n+1)$-tuple is assigned to a unique arc. With this labeling we can inductively construct the digraph B_{n+1}. We include an arc from $\mathbf{x} \in B^{n+1}$ to $\mathbf{y} \in B^{n+1}$ in this new graph B_{n+1} whenever the terminal vertex of the arc $\mathbf{x} \in B_n$ is the same as the initial vertex of the arc $\mathbf{y} \in B_n$. We see that a vertex $\mathbf{x}$ is adjacent to a vertex $\mathbf{y}$ in the induced digraph when the last n coordinates of x match the first n coordinates of $\mathbf{y}$ in the respective $(n+1)$-tuples representing $\mathbf{x}$ and $\mathbf{y}$. This induced digraph is isomorphic to B_{n+1} by the identity mapping. Thus, a closed Eulerian trail of length 2^{n+1} visiting every arc in B_n defines a Hamiltonian cycle visiting every vertex in B_{n+1}. Since $(0,1)$ is a Hamiltonian cycle in B_1, and there exists a Eulerian trail in B_n, $n \geq 1$, we conclude that a Hamiltonian cycle exists in B_n for $n \geq 1$.

Another way to label the arcs in the digraph B_n is to simply label the arc going from the vertex $(x_0, \ldots, x_{n-1})$ to the vertex $(x_1, \ldots, x_n)$ with the single bit x_n. The set of labels encountered as we trace the arcs along a Hamiltonian cycle (path) in B_n generates a binary *de Bruijn cycle (sequence)*. Such a cycle is periodic of period 2^n and contains each of the 2^n different binary n-tuples exactly one time in each period of the sequence. The set of all de Bruijn cycles of length 2^n is denoted as S_n.

A subsequence of n consecutive terms (or an *n-sequence*) from a sequence $S = (s_1, s_2, \ldots, s_{2^n})$ is a string of the form $(s_i, s_{i+1}, \ldots, s_{i+n-1})$, where we use the convention that the subscripts are taken modulo 2^n. That is, we allow an n-sequence

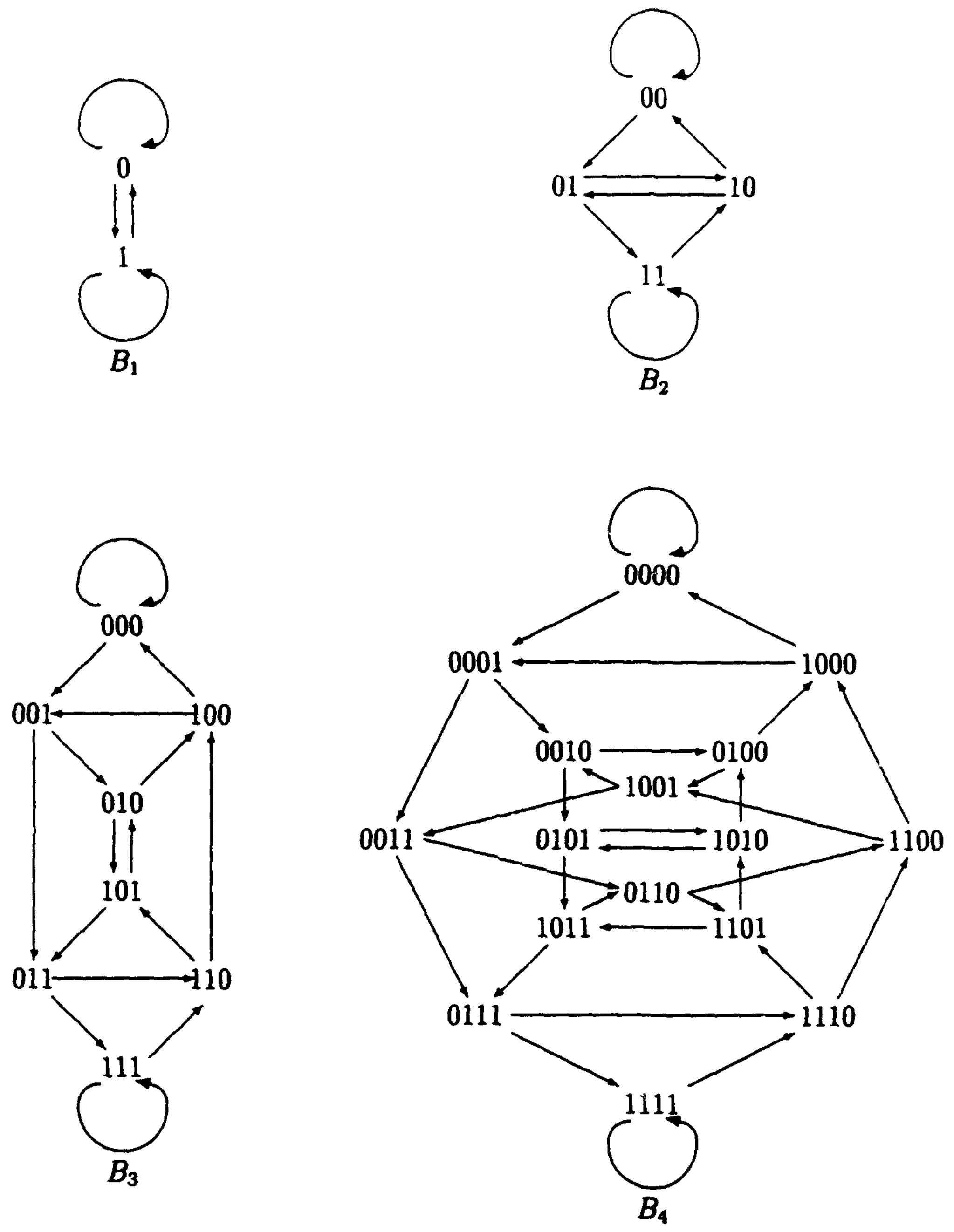


Figure 7. *Good - de Bruijn graphs* B_n, $1 \leq n \leq 4$

of the form

$$(s_{2^n-j}, s_{2^n-j+1}, \ldots, s_{2^n}, s_1, s_2, \ldots, s_{n-j-1}), \qquad j = 0, 1, \ldots, n-2,$$

which "wraps around" from the end of S to the beginning. Any circular permutation of S is considered to be the same sequence.

A cycle of period p in B_n is an ordered set of distinct vertices (states) $(v_0, v_1, \ldots, v_{p-1})$, such that v_{i+1} is the successor of v_i, i.e., $v_{i+1} = F(v_i)$, $i = 0, 1, \ldots, p-2$, and $v_0 = F(v_{p-1})$. Clearly, an edge-factor in B_n is isomorphic to a factor in B_{n+1}.

In addition to having two successors, each vertex in B_n has exactly two predecessors. Moreover, if vertex $\mathbf{x}$ is a predecessor of vertex $\mathbf{y}$, the following is always true:

1. $\hat{\mathbf{x}}$ is a predecessor of $\mathbf{y}$.
2. $\mathbf{x}$ is a predecessor of $\mathbf{y}'$.
3. $\hat{\mathbf{x}}$ is a predecessor of $\mathbf{y}'$.

The four vertices $\mathbf{x}$, $\hat{\mathbf{x}}$, $\mathbf{y}$, and $\mathbf{y}'$ are commonly called an *adjacency quadruple*. An arc $(\mathbf{x}, \mathbf{y})$ is directed from vertex $\mathbf{x}$ to vertex $\mathbf{y}$ and called *incident* with both $\mathbf{x}$ and $\mathbf{y}$. Conversely, $\mathbf{x}$ and $\mathbf{y}$ are incident to the arc $(\mathbf{x}, \mathbf{y})$. A vertex $\mathbf{x} = (x_0, x_1, \ldots, x_{n-1})$ in B_n is incident to the arcs $(0, x_0, x_1, \ldots, x_{n-1})$, $(1, x_0, x_1, \ldots, x_{n-1})$, $(x_0, x_1, \ldots, x_{n-1}, 0)$, and $(x_0, x_1, \ldots, x_{n-1}, 1)$ that constitute an adjacency quadruple in B_{n+1}.

In an edge-factor of B_n, each arc uniquely identifies its cycle. The cycle to which the arc α belongs is denoted by (α). The set of incident vertices for a given cycle (α) is denoted by $(\alpha)_v$. Two cycles (α) and (β) are *adjacent* if they are arc disjoint and there exists an arc γ on (α) whose conjugate $\hat{\gamma}$ is on (β) (i.e., α, β are both incident to same vertex v). It follows that (α) and (β) are adjacent whenever (α) and (β) are disjoint and $(\alpha)_v \cap (\beta)_v \neq \emptyset$. Consequently, each vertex is incident to 2 distinct cycles in an edge-factor consisting of simple cycles.

Furthermore, let ${}_y(\alpha)_v$ denote the set of vertices $v_i \in (\alpha)_v$ for which $d(y, v_i) = d(y, (\alpha)_v)$. From each set ${}_y(\alpha)_v$ a specific vertex $v_i \in {}_y(\alpha)_v$ is designated as the *representative vertex* of (α), denoted by $v_i = {}_y(\alpha)_R$.

Recall that the *distance*, $d(x,y)$, between two vertices x and y in a weak digraph is the minimum length of a walk joining them in the underlying graph. From Equation II.1 on page 16, the distance, $d(y, (\alpha)_v)$, between a vertex $\mathbf{y}$ and the set of vertices $(\alpha)_v$ in B_n equals $\min\limits_{x \in (\alpha)_v} \{d(\mathbf{y}, x)\}$.

One factor occurs so often that it has been given a special name. The *Pure Cycling Register Factor* consists of cycles formed by the cyclic rotation of the bits in the Pure Cycling Register, PCR_n. Golomb [Ref. 6] shows that the number of cycles, $Z(n)$, in the PCR_n is given by $Z(n) = \frac{1}{n}\sum\limits_{d|n} \phi(d) 2^{\frac{n}{d}}$, where ϕ is Euler's totient function and the summation is over the divisors of n. See Table I for values of $\phi(n)$ and $Z(n)$. Mykkeltveit [Ref. 20] shows that no shift register (linear or nonlinear) of length n can generate a factor containing more than $Z(n)$ cycles.

n	$\phi(n)$	$Z(n)$
1	1	2
2	1	3
3	2	4
4	2	6
5	4	8
6	2	14
7	6	20
8	4	36
9	6	60
10	4	108
11	10	188
12	4	352

Table I. *Values of Euler's function* $\phi(n)$ *and of the cycle function* $Z(n)$, $1 \leq n \leq 12$

III. COMBING SEQUENCES

> *"I want a clean cup," interrupted the Hatter. "Let's all move one place on." He moved on as he spoke, and the Dormouse followed him. The March Hare moved into the Dormouse's place and Alice rather unwillingly took the place of the March Hare. The Hatter was the only one who got any advantage from the change.*
>
> Alice's Adventures in Wonderland, Chapter 7

A. INTRODUCTION

In Chapter I , Section C, we introduced the functional $\mathcal{F}(f_1, f_2, \ldots, f_{2^n})$, whose domain consists of the 2^n projections from B^l to B^n defined on a sequence of period 2^n by

$$f_i(s_i, s_{i+1}, \ldots, s_{i+l-1}) = (s_i, s_{j_1}, \ldots, s_{j_{n-2}}, s_{i+l-1}),$$

where $i < j_1 < \cdots < j_{n-2} < i + l - 1$.

The function f_i can be viewed as an l-long window containing n viewing stations or as comb of length l with n surviving teeth, which we call an (l, n)-comb. Occasionally, we categorize such a comb by its longest string of consecutive teeth. An m-comb denotes a comb with at most m consecutive teeth. It is also sometimes convenient to normalize a comb so that a longest string of consecutive teeth always appears on the left. For example, the (5,3)-comb, $\underline{|\bullet\bullet||}$, defined on a 2^3-long sequence by

$$f_i(s_i, s_{i+1}, s_{i+2}, s_{i+3}, s_{i+4}) = (s_i, s_{i+3}, s_{i+4}),$$

is normalized to the (6,3)-comb, $\underline{||\bullet\bullet\bullet|}$, specified by

$$f_i(s_i, s_{i+1}, s_{i+2}, s_{i+3}, s_{i+4}, s_{i+5}) = (s_i, s_{i+1}, s_{i+5}),$$

i.e., $\underline{|\bullet\bullet||} \xrightarrow{\text{normalized}} \underline{||\bullet\bullet\bullet|}$. (Each are 2-combs). There also may be instances when one wants to use the version of the comb that has the smallest length l.

If a sequence, S_n, of length 2^n has the property that some (l, n)-comb can extract every different n-tuple, then S_n is called *complete* or *nonclassical de Bruijn*. It follows that the reverse and complement of a complete sequence are also complete. By definition, every de Bruijn cycle is complete.

B. SEQUENCES AND THEIR EXTRACTED n-TUPLES

To illustrate these concepts, consider the sequence S of length 2^4 given by S = 0000111100101101. Using the classical de Bruijn (4,4)-comb denoted by the symbol, $C = \underline{||||}$, and specified by

$$f_i(s_i, s_{i+1}, s_{i+2}, s_{i+3}) = (s_i, s_{i+1}, s_{i+2}, s_{i+3}),$$

4-tuples are extracted from S in the following order: f_1 → 0000, f_2 → 0001, f_3 → 0011, f_4 → 0111, f_5 → 1111, f_6 → 1110, f_7 → 1100, f_8 → 1001, f_9 → 0010 f_{10} → 0101, f_{11} → 1011, f_{12} → 0110, f_{13} → 1101, f_{14} → 1010, f_{15} → 0100, f_{16} → 1000. The (4,4)-comb applied to S generates a sequence $T(C, S)$, of 4-tuples that specifies a Hamiltonian cycle along the Good - de Bruijn graph B_4. Figure 8 represents this cycle. We see that the sequences, S and $T(C, S)$, define the same cycle around B_n.

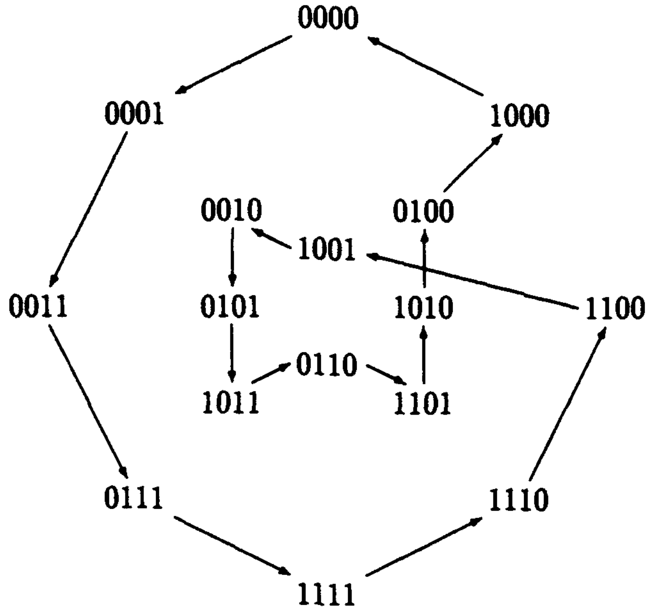


Figure 8. *A Hamiltonian cycle in the Good - de Bruijn digraph* B_4

The (8,4)-comb, $C' = \underline{|||\bullet\bullet\bullet\bullet|}$, specified by

$$f_i(s_i, s_{i+1}, s_{i+2}, s_{i+3}, s_{i+4}, s_{i+5}, s_{i+6}, s_{i+7}) = (s_i, s_{i+1}, s_{i+2}, s_{i+7}),$$

generates the following sequence, $T(C', S)$, of distinct 4-tuples from S: (0001, 0000, 0010, 0111, 1110, 1111, 1101, 1000, 0011, 0100, 1010, 0110, 1100, 1011, 0101, 1001). The sequence $T(C', S)$, however, does not represent a Hamiltonian cycle along the de Bruijn digraph B_4 since there are consecutive 4-tuples in $T(C', S)$ that are not adjacent in B_4. Since this (8,4)-comb has 3 consecutive teeth (i.e., it is a 3-comb), the sequence S *must* specify a walk along the graph B_3 (Figure 9A) that visits every

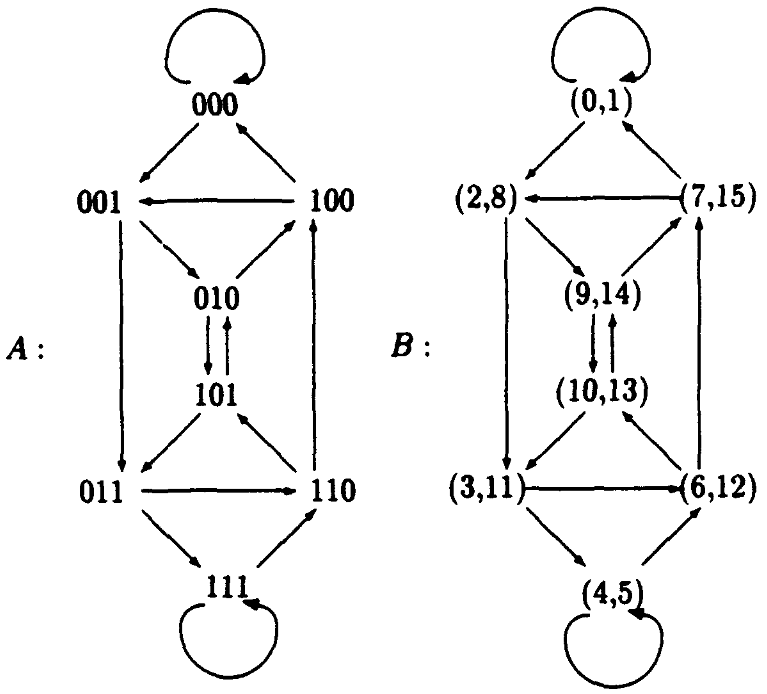


Figure 9. *Good - de Bruijn digraph B_3*

vertex exactly twice. The visitation times for each vertex in this walk are shown in the parentheses in Figure 9B.

Let C be an 3-comb for the sequence S. It follows immediately from de Bruijn's doubling theorem [Ref. 13] that the sequence S, since it visits every vertex of B_3 twice, also induces a double Eulerian cycle along the digraph B_2. Furthermore, the sequence $T(C, S)$, of 4-tuples specifies a double Eulerian cycle along $\overline{B}_{4-(1+1)}$, a digraph isomorphic to B_2 [See Appendix B]. These two respective cycles, with visitation times in parentheses, are shown in Figure 10. The sequences $T(C, S)$ and S define identical walks along the isomorphic graphs $\overline{B}_{4-(1+1)}$ and B_2, respectively.

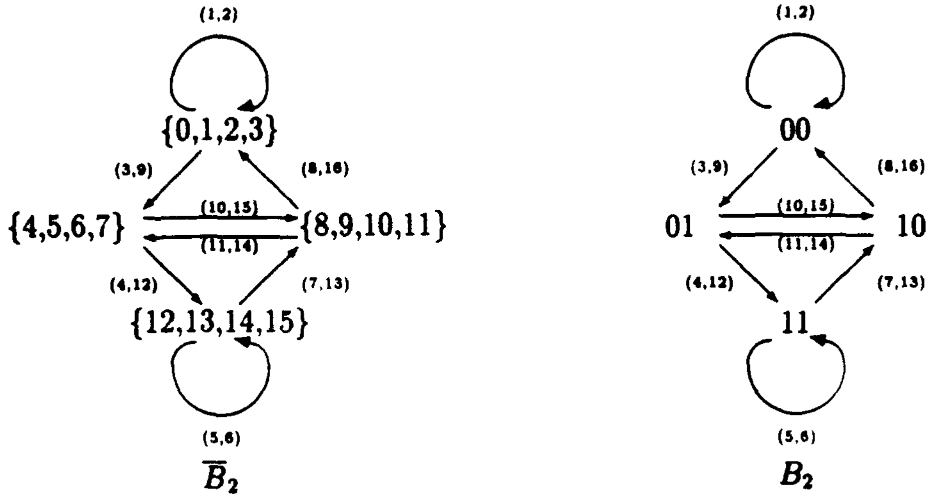


Figure 10. $\overline{B}_{4-(1+1)}$ *and the Good - de Bruijn digraph* B_2

Lemma III.1 *Let S_n be a complete binary sequence of length 2^n with an associated (n-k)-comb. Then S_n specifies a walk that traverses each arc along the digraph $B_{n-(k+1)}$ exactly 2^k times where $(k+1) < n$.*

Proof: Since S_n is complete, each $(n-k)$-tuple occurs exactly 2^k times. The arcs in the Good - de Bruijn digraph, $B_{n-(k+1)}$, are uniquely represented by $(n-k)$-tuples. Therefore, S_n, specifies a walk that traverses each arc exactly 2^k times along the digraph $B_{n-(k+1)}$. ■

Corollary III.2 *Let S_n be a complete binary sequence of length 2^n with an associated (n-k)-comb, C. Then the sequence, $T(C, S_n)$, of n-tuples extracted by the comb specifies a walk that traverses each arc exactly 2^k times along the digraph $\overline{B}_{n-(k+1)}$ where $(k+1) < n$.*

Proof: Follows immediately from Lemma III.1 and the definition of the isomorphic digraph $\overline{B}_{n-(k+1)}$. ■

The elements of the set $\mathcal{W}_{2^k}$ for $B_{n-(k+1)}$ define 2^n-long walks along the arcs of the graph $B_{n-(k+1)}$ where each arc is visited exactly 2^k times. The set $\mathcal{W}_{2^k}$ includes all the sequences having an $(n-k)$-comb. The following theorem of van Aardenne-Ehrenfest and de Bruijn [Ref. 12] allows us to count, in any Eulerian regular directed graph D, the number of sequences that define a walk along D where each arc is traversed exactly ρ times. Table II provides the number of sequences defining an Eulerian or double Eulerian cycle along B_n, for $1 \leq n \leq 4$.

Theorem III.3 (van Aardenne-Ehrenfest and de Bruijn) *Let $D = (V, A)$ be an Eulerian regular directed graph where in(v) $= \sigma$. Then the number of ways, $\|D\|_\rho$, to traverse each arc in D exactly ρ times is*

$$\|D\|_\rho = \frac{1}{\rho}\sum_{d|\rho}\phi\left(\frac{\rho}{d}\right)\left(\frac{(\sigma d)!}{(d!)^\sigma\,\sigma!}\right)^{|V|}\|D\|_1 \qquad \text{(III.1)}$$

where ϕ is the Euler's totient function, $\|D\|_1$ is the number of Eulerian cycles in D, and the summation is extended over all divisors of ρ.

Theorem III.3 was proved using an interesting but considerably complicated argument. The development and proof of the theorem is summarized in Appendix A.

Let S be a binary sequence of length 2^n with an $(n-1)$-comb. By Lemma III.1, S is an element in the set of sequences, $\mathcal{W}_2$, that specify a double Eulerian cycle along

n	Eulerian	Double Eulerian
1	1	5
2	2	82
3	16	52,496
4	2048	44,079,843,328

Table II. *The number of Eulerian and double Eulerian cycles along B_n,* $1 \leq n \leq 4$

B_{n-2}. From Theorem III.3, the number of distinct double Eulerian cycles along B_{n-2} is given by

$$\|B_{n-2}\|_2 = (1+3^{2^{n-2}})2^{2^{n-3}-n+1}. \qquad \text{(III.2)}$$

Not every sequence, however, in $\mathcal{W}_2$ has an $(n-1)$-comb. Finding the general formula for the number of sequences with an $(n-1)$-comb has been elusive. In the following two sections we show that the number of sequences of length 2^n possessing an $(n-1)$-comb is much smaller than the number of sequences with an n-comb.

C. THE RUN STRUCTURE OF SEQUENCES POSSESSING (n-1)-COMBS

It is well-known that the number of de Bruijn cycles of length 2^n is $2^{2^{n-1}-n}$. Therefore, of all of the double Eulerian cycles along B_{n-2}, there are $2^{2^{n-1}-n}$ that are defined by de Bruijn cycles. In this section we show that, statistically, we should expect the number of 2^n-long sequences with an $(n-1)$-comb to be only $2^{2^{n-2}-n+2}$.

A *run of ZEROs* in a binary sequence is defined to be a subsequence of consecutive 0's that is preceeded and followed by a 1. The *length* of a run of ZEROs is the number of consecutive 0's in the subsequence. A *run of ONEs* is similary defined. A binary sequence can be interpreted as a sequence of integers representing the lengths of the alternating runs of ZEROs and ONEs in the sequence. For example, the sequence:

$$S = 00001010110111010011001000011111$$

corresponds to the run sequence $R = (4111121311222145)$. The sequence, R, consists of a subsequence $\mathcal{Z}_5 = (41111224)$ of the run lengths of runs of ZEROs interleaved (or perfectly shuffled [Ref. 21]) with a subsequence $\mathcal{O}_5 = (11231215)$ of the run lengths of runs of ONEs.

The multiset consisting of the lengths of the runs of ZEROs and ONEs in a de Bruijn cycle is completely determined, (See Chapter V or [Ref. 6]). In fact, a run sequence that defines a cycle in $\mathcal{W}_\gamma$ in B_n can also be determined. For example, when S is a sequence that defines a double Eulerian cycle along B_n, the multiset of the lengths of the runs of ZEROs or ONEs in S consists of one of only two possible multisets, respectively.

Theorem III.4 *Let S be a sequence defining a double Eulerian cycle along B_n. Then S satisfies one of the following 2 criteria:*

1. *S has 2^{n-k} runs of ZEROs of length k for $1 \leq k \leq n$ and a single run of ZEROs of length $n+2$, or*
2. *S has 2^{n-k} runs of ZEROs of length k for $1 \leq k \leq n-1$ and two runs of ZEROs of length $n+1$.*

The same distribution holds for the runs of ONEs.

Proof: Since each $(n+1)$-tuple occurs exactly twice in S, the longest runs of ZEROs must consist of either:

1. Exactly 1 run of ZEROs of length $n+2$, or
2. Exactly 2 runs of ZEROs of length $n+1$.

Case 1: The run of ZEROs of length $n+2$ must be preceded and followed by a 1, or the $(n+1)$-tuple $(00\cdots00)$ would appear at least three times in S. The $(n+1)$-tuple consisting of a 1 followed by the n-tuple $(00\ldots0)$ also occurs exactly twice in the sequence. One occurrence, however, is already accounted for by the run of ZEROs of length $n+2$. Thus, there is an additional run of ZEROs of length n that provides the second $(n+1)$-tuple consisting of a 1 followed by the n-tuple $00\ldots0$. Thus, there is no run of ZEROs of length $n+1$. To find the number of runs of ZEROs of length k, for $1 \le k \le n-1$, we consider all $n+1$ consecutive bits of the sequence that begin with a 1 followed by the k-tuple $00\ldots0$ and then a 1. Each such run can be made to correspond to an arbitrary $(n+1)$-tuple of the form

$$1\underbrace{00\ldots0}_{k}1\underbrace{xx\ldots x}_{n-k-1},$$

where the x's are chosen as arbitrary bits. Since we are free to choose each of the remaining $n-k-1$ bits, there are $2^{n-k-1} \times 2$ runs of ZEROs of length k for $1 \le k \le n-1$. With the single run of ZEROs of length n and the single

n	Eligible Run Distributions for B_n
1	$\{3,1\}$
	$\{2,2\}$
2	$\{4,2,1,1\}$
	$\{3,3,1,1\}$
3	$\{5,3,2,2,1,1,1,1\}$
	$\{4,4,2,2,1,1,1,1\}$
4	$\{6,4,3,3,2,2,2,2,1,1,1,1,1,1,1,1\}$
	$\{5,5,3,3,2,2,2,2,1,1,1,1,1,1,1,1\}$

Table III. *Distribution of runs for double Eulerian cycles along B_n*

run of ZEROs of length $n + 2$ the result follows. The same argument holds for runs of ONEs.

Case 2: The two runs of ZEROs of length $n + 1$ must each be preceded and followed by a 1, or the $(n + 1)$-tuple, $00 \cdots 00$, would appear at least three times in S. The $(n + 1)$-tuple consisting of a 1 followed by the n-tuple $00 \ldots 0$ occurs exactly twice in the sequence. These, however, are already accounted for by the two runs of ZEROs of length $n + 1$. Thus, there is no run of ZEROs of length n. In a like manner as above, there are $2^{n-k-1} \times 2$ runs of ZEROs of length k, for $1 \leq k \leq n - 1$. With the two runs of ZEROs of length $n + 1$ the result follows. The same argument holds for the run distribution of ONEs. ∎

Every double Eulerian cycle around B_n can be described by a binary sequence whose run sequence, R, consists of a subsequence $\mathcal{Z}_n$ interleaved with a subsequence $\mathcal{O}_n$. The subsequences $\mathcal{Z}_n$ and $\mathcal{O}_n$ are respectively a permutation of one of the two multisets in Theorem III.4. To determine how many sequences have $(n - 1)$-combs, it is useful to partition all of the double Eulerian cycles into one of the following 3 sets: $\mathcal{P}_{(n,2)}$, $\mathcal{P}_{(n,1)}$, and $\mathcal{P}_{(n)}$ defined as:

1. The set, $\mathcal{P}_{(n,2)}$, consists of sequences where both the ZERO and ONE run sequences are permutations of the multiset that includes the element $n+2$.
2. The set, $\mathcal{P}_{(n,1)}$, consists of sequences where both the ZERO and ONE run sequences are permutations of the multiset that includes the element $n+1$.
3. The set, $\mathcal{P}_{(n)}$, consists of sequences where the ZERO and ONE run sequences are permutations of different multisets.

The sizes of each of the 3 sets, $\mathcal{P}_{(n,2)}$, $\mathcal{P}_{(n,1)}$, and $\mathcal{P}_{(n)}$, are determined in a straightforward manner. Equation III.2 provides the total number of double Eulerian cycles, $\|B_n\|_2$, where $\|B_n\|_2 = |\mathcal{P}_{(n,2)}| + |\mathcal{P}_{(n,1)}| + |\mathcal{P}_{(n)}|$. Let $\hat{B}_n$ denote the Eulerian digraph constructed from the Good - de Bruijn digraph by removing the two arcs (loops), (0) and (1). One can see that $|\mathcal{P}_{(n,1)}|$ is equivalent to the number of sequences representing double Eulerian cycles along $\hat{B}_n$, i.e., $|\mathcal{P}_{(n,1)}| = \|\hat{B}_n\|_2$. From Equation A.5 in Appendix A we find

$$\begin{aligned} |\mathcal{P}_{(n,1)}| &= \tfrac{1}{2}(2^{(2^n-n-1)} + 2^{-(2^{n+1}-2)}2^{(2^n-n-1)}12^{(2^n-2)}2^2) \\ &= 2^{(2^n-n-2)} + 2^{(2^n-n-2)}3^{(2^n-2)} \qquad \text{(III.3)} \\ &= 2^{(2^n-n-2)}(1+3^{(2^n-2)}). \end{aligned}$$

Now let $\tilde{B}_n$ and $\ddot{B}_n$ denote Eulerian digraphs constructed from the Good - de Bruijn digraph by removing just the loop, (0) and (1), respectively. We define $\mathcal{Q}$ to be the set of sequences representing double Eulerian cycles along $\tilde{B}_n$. By inserting a 0 into the longest run of ZEROs of a sequence $Q \in \mathcal{Q}$, a sequence is created that contains a run distribution of ZEROs that includes the element $(n+1)$ twice and a run distribution of ONEs from either of the two eligible multisets. In a similar manner, each sequence representing double Eulerian cycles along $\ddot{B}_n$ is equivalent to a sequence with a run distribution of ONEs that includes the element $(n+2)$ and a run distribution of ZEROs from either of the two eligible multisets. We define $\mathcal{R}$ to be set of sequences representing the double Eulerian cycles along $\ddot{B}_n$. The set $\mathcal{Q} \cup \mathcal{R} = \mathcal{U}$ is equivalent to the set of double Eulerian cycles that includes all the elements of $\mathcal{P}_{(n)}$. In addition to the elements of $\mathcal{P}_{(n)}$, the set $\mathcal{U}$ also includes the sequences that have ZERO and ONE run sequences appearing as permutations of the multiset that includes the element $n+1$, i.e., $\mathcal{P}_{(n,1)} \cup \mathcal{P}_{(n)} = \mathcal{U}$. It follows immediately that

$$|\mathcal{P}| = \left(\|\tilde{B}_n\|_2 - |\mathcal{P}_{(n,1)}|\right) \times 2,$$

since $\|\tilde{B}_n\|_2 = \|\ddot{B}_n\|_2$. By again applying Equation A.5 in Appendix A, we find that

$$\begin{aligned} |\mathcal{P}_{(n)}| &= \left(\tfrac{1}{2}(2^{(2^n-n-1)} + 2^{-(2^{n+1}-1)}2^{(2^n-n-1)}12^{(2^n-1)}2^1) - 2^{(2^n-n-2)}(1+3^{(2^n-2)})\right) \times 2 \\ &= \left(2^{(2^n-n-2)} + 2^{(2^n-n-2)}3^{(2^n-1)} - 2^{(2^n-n-2)}(1+3^{(2^n-2)})\right) \times 2 \\ &= 2^{(2^n-n)}3^{(2^n-2)}. \end{aligned} \tag{III.4}$$

Consequently, it follows that

$$\begin{aligned} |\mathcal{P}_{(n,2)}| &= \|B_n\|_2 - |\mathcal{P}_{(n,1)}| - |\mathcal{P}_{(n)}| \\ &= 2^{(2^n-n)}3^{(2^n-2)}. \end{aligned} \tag{III.5}$$

Table IV provides the values for $\|B_n\|_2$, $|\mathcal{P}_{(n,2)}|$, $|\mathcal{P}_{(n,1)}|$, and $|\mathcal{P}_{(n)}|$, for $1 \leq n \leq 4$.

n	$\|\|B_n\|\|_2$	$\|\mathcal{P}_{(n,2)}\|$	$\|\mathcal{P}_{(n,1)}\|$	$\|\mathcal{P}_{(n)}\|$
1	5	2	1	2
2	82	36	10	36
3	52,496	23,328	5,840	23,328
4	44,079,843,328	19,591,041,024	4,897,761,280	19,591,041,024

Table IV. *Double Eulerian cycles along B_n, by category*

D. RANDOMNESS OF THE DOUBLE EULERIAN WALKS

In this section we show that the number of classical de Bruijn cycles in the set $\mathcal{P}_{(n,2)}$ coincides with the expected number of de Bruijn cycles that would exist if the sequences in $\mathcal{P}_{(n,2)}$ had a particular randomness property. (Recall that of the $2^{(2^n-n)}3^{(2^n-2)}$ sequences in $\mathcal{P}_{(n,2)}$, exactly $2^{2^{n+1}-n-2}$ are de Bruijn, since the sequence length corresponds to a Hamiltonian path through B_{n+2}.)

A Randomness Property: From any vertex, each departing arc is *equally likely* to be traversed next in a walk along B_n defined by the 2^{n+2}-long sequence, S_{n+2}, where $S_{n+2} \in \mathcal{P}_{(n,2)}$.

Each vertex in B_n is entered 4 times along the walk defined by the sequence S_{n+2}. Let the arcs in B_n be labeled with 0's and 1's as described in Chapter II. Suppose vertex v is entered from the arc labeled α_1 for the first time. Let X_1 be a binary indicator variable where $X_1 = 1$ if and only if the next arc traversed from v is labeled 1. The indicator variable $X_1 = 0$ if and only if the next arc traversed from v is labeled 0. Similarly, when a vertex v is entered from the arc labeled α_1 for the second time, let X_2 be a binary indicator variable where $X_2 = 1$ if and only if the next arc traversed from v is labeled 1. The indicator variable $X_2 = 0$ if and only if the next arc traversed from v is labeled 0.

The joint probability function for the two discrete random variables X_1 and X_2 is given by

$$p_{X_1,X_2}(x_1,x_2) = P(X_1 = x_1, X_2 = x_2), \text{ for } x_1, x_2 \in \{0,1\}.$$

If $X_1 \neq X_2$ for each vertex in B_n, the sequence S_{n+2} is de Bruijn of length 2^{n+2}, since each $(n+2)$-tuple is distinct. The probability that $X_1 \neq X_2$ for each vertex in B_n for the walk S_{n+2} can be computed quite easily.

Let v be a vertex that is not incident with a loop in B_n. Since S_{n+2} defines a walk where each outgoing arc from v is equally likely to be visited next and each arc in B_n *is visited exactly two times, it follows that*

$$P(X_2 = 1 | X_1 = 0) = \frac{2}{3},$$

where the bar within the parentheses indicates conditional probability. Furthermore, if v is a vertex incident with a loop then,

$$P(X_2 = 1 | X_1 = 0) = 1,$$

since each arc in B_n must be visited twice and we must visit the loop at this time or miss it entirely. Therefore, the probability that S_{n+2} is de Bruijn is $p = \left(\frac{2}{3}\right)^{2^n - 2}$.

If the sequences in $\mathcal{P}_{(n,2)}$ have the property that they each define a walk by which subsequent arcs from each vertex are equally likely to be traversed, we would expect

$$|\mathcal{P}_{(n,2)}| \times \left(\frac{2}{3}\right)^{2^n-2} = 2^{2^n-n}3^{2^n-2} \times \left(\frac{2}{3}\right)^{2^n-2} = 2^{2^{n+1}-n-2}$$

sequences in $\mathcal{P}_{(n,2)}$ to possess the de Bruijn property. Since there are exactly $2^{2^{n+1}-n-2}$ de Bruijn cycles in $\mathcal{P}_{(n,2)}$, this expectation is realized. In the next section, we discuss the implication of this randomness property with respect to the expected number of complete cycles associated with an $(n-1)$-comb.

E. THE NUMBER OF COMPLETE CYCLES

Counting the exact number of sequences associated with an $(n-k)$-comb, $1 \leq k \leq 2^{n-1}-1$, remains a difficult problem. We can, however, estimate the size of the set of complete cycles associated with a particular comb. One could postulate that there are equal numbers of sequences for each possible comb, especially since the previous section suggests that the successive bits (0 and 1) are equally likely among the set of eligible sequences. It has been found by a computer search that sequences with n-combs tend to be much more common than sequences with $(n-k)$-combs, for $k \geq 1$. This result is supported by the same probabilistic argument developed in the previous section.

For simplicity, we again utilize the sequence S_{n+2} defined in Section D. Here we find the probability that S_{n+2} supports an $(n-1)$-comb of the type $\underline{\overbrace{\| \cdots \|}^{n-1} \bullet |}$. Suppose that vertex v is entered from the arc labeled α_1 for the first time. Then, the indicator variables X_1 and X_2 are defined as before, except that now we are concerned with the *second* arc traversed from v. Therefore, we again let X_1 be a binary indicator variable where $X_1 = 1$ if and only if the *second* arc traversed from v is labeled 1. The indicator variable $X_1 = 0$ if and only if the second arc traversed from v is labeled 0. Similarly, when vertex v is entered from the arc labeled α_1 for the second time, let X_2 be a binary indicator variable where $X_2 = 1$ if and only if the second arc traversed from v is labeled 1. The indicator variable $X_2 = 0$ if and only if the second arc traversed from v is labeled 0.

In contrast to the situation described in Section D, the inequality of X_1 and X_2 for all vertices in B_n is not a sufficient condition for S_{n+2} to support an $(n-1)$-comb. It must also be the case that $Y_1 \neq Y_2$, where Y_1 and Y_2 are similarly defined on the *other* arc, α_2, entering the vertex v. Therefore, when vertex v is entered from the arc labeled α_2 for the first time, let Y_1 be a binary indicator variable where $Y_1 = 1$ if and only if the second arc traversed from v is labeled 1. The indicator variable $Y_1 = 0$ if and only if the second arc traversed from v is labeled 0. Similarly, when vertex v is entered from the arc labeled α_2 for the second time, let Y_2 be a binary indicator variable where $Y_2 = 1$ if and only if the second arc traversed from v is labeled 1. The indicator variable $Y_2 = 0$ if and only if the second arc traversed from v is labeled 0.

The joint probability function for the 4 discrete random variables X_1, X_2, Y_1, and Y_2 is given by

$$p_{X_1,X_2,Y_1,Y_2}(x_1, x_2, y_1, y_2) = P(X_1 = x_1, X_2 = x_2, Y_1 = y_1, Y_2 = y_2),$$

for $x_1, x_2, y_1, y_2 \in \{0, 1\}$. If $X_1 \neq X_2$ and $Y_1 \neq Y_2$ for all the vertices in B_n, the sequence has a comb specified by $\underline{\overbrace{\| \cdots \|}^{n-1} \bullet |}$. The probability that $X_1 \neq X_2$ for each vertex in B_n not incident with a loop is $\frac{5}{9}$. Given that $X_1 \neq X_2$, the probability that $Y_1 \neq Y_2$ is $\frac{2}{3}$. Therefore the probability of $X_1 \neq X_2$ and $Y_1 \neq Y_2$ is $\frac{10}{27} \approx \frac{1}{3}$.

Furthermore, if v is a vertex incident with a loop then,

$$P(X_2 = 1 | X_1 = 0) = 1,$$

since each arc must be visited twice. Therefore, the probability that S_{n+2} has a $\underline{\overbrace{\| \cdots \|}^{n-1} \bullet |}$ comb is *approximately* $\left(\frac{1}{3}\right)^{2^n-2}$. We would expect to find that

$$|\mathcal{P}_{(n,2)}| \times \left(\frac{1}{3}\right)^{2^n-2} = 2^{2^n-n}3^{2^n-2} \times \left(\frac{1}{3}\right)^{2^n-2} = 2^{2^n-n}$$

of the sequences in $\mathcal{P}_{(n,2)}$ have these combs. For $n = 3$, of the 23,328 sequences in the set $\mathcal{P}$, there are exactly 2^5 combs of the type $\underline{||||\bullet|}$, supporting our claim. In general, the experimental results support the probabilistic predictions. We have found that the sizes of the set of 2^n-long sequences with $(n-1)$-combs, of all types, is substantially smaller than $2^{2^{n-1}-n}$, the number of de Bruijn sequences of length 2^n.

In the following chapters we develop the necessary theoretical concepts on double Eulerian cycles to analyze the Good - de Bruijn digraphs and to gain a better understanding of complete cycles. A complete combinatorial explanation for the number of sequences satisfying a particular comb is an intriguing goal that seems very hard to achieve at present.

IV. A MEASURE ON A GRAPH

"Would you tell me, please, which way I ought to go from here?"
"That depends a good deal on where you want to get to," said the cat.
" I don't much care where ..." said Alice
"Then it doesn't matter which way you go," said the cat.

Alice's adventures in Wonderland, Chapter 7

A. INTRODUCTION

In this chapter we develop the theoretical concepts regarding double Eulerian cycles, i.e., walks on a connected graph or weak digraph that visit every edge exactly twice. A measure can be given to the visitation pattern of a double Eulerian cycle.

Every de Bruijn cycle (and some nonclassical de Bruijn cycles) of length 2^n defines a walk that visits every vertex on B_n, traverses each arc of B_{n-1}, and passes through each arc on B_{n-2} exactly twice. From these latter walks emerges a measure of complexity of the sequence in question and a notion of the *value* for the underlying graph.

B. THE VALUE OF A GRAPH

We define $\mathcal{W}_2$ to be the set of walks that traverses each edge (arc) in the graph (digraph) exactly twice. The set $\mathcal{W}_2$ is called the set of *double Eulerian cycles.* We define a *measure* on a walk in $\mathcal{W}_2$ as the sum, over all edges of the graph, of the positive difference of the visitation times on each edge. The *value* of a graph G is

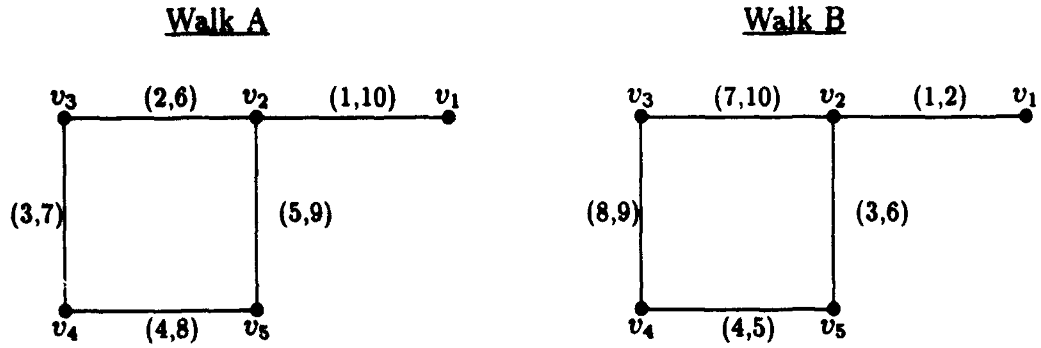


Figure 11. *Double Eulerian cycles with visitation times*

then defined to be the minimum measure over all double Eulerian walks on G. In addition, every connected graph always has a value.

In Figure 11, two different double Eulerian walks are presented. The numbers in parentheses are the visitation times. Walk $A = (v_1, v_2, v_3, v_4, v_5, v_2, v_3, v_4, v_5, v_2, v_1)$ has a measure of 25. Walk $B = (v_2, v_1, v_2, v_5, v_4, v_5, v_2, v_3, v_4, v_3, v_2)$ has a measure of 9, which can be shown to be the value of the graph.

Theorem IV.1 *Let G be a connected graph. Then G has at least one double Eulerian walk.*

Proof: Let H be the graph generated by duplicating each edge in G. The degree of each vertex in H is then even. Therefore, Theorem II.1 implies that H has a closed Eulerian trail and G must have a double Eulerian cycle. As an immediate corollary, a double Eulerian cycle in G yields an Eulerian walk in H. ■

Corollary IV.2 *Every double Eulerian cycle in a grapı. G is a closed walk.*

Proof: Let H be the graph determined by duplicating each edge in G. From Theorem IV.1, a walk in H is Eulerian if and only if the walk is double Eulerian in G. The degree of each vertex in H can be counted by following an Eulerian walk W_1 in H. Each occurrence of a vertex along W_1 adds 2 to the degree of an interior vertex in W_1 and 1 to the degree of the initial and terminal vertices (recall that a walk is a sequence of vertices and edges). Since the degree of each vertex in H is even, the initial and terminal vertices of the walk must be the same vertex. Therefore, every double Eulerian cycle in G is closed. ■

Since a double Eulerian cycle is closed, the edge sequence representing a double Eulerian cycle can be viewed as a cycle. Alternatively, we can also define a measure on a walk in $\mathcal{W}_2$ as the sum over all edges of the distance along the cycle between the two occurrences of each edge. From this perspective, the cycle distance between identical edges in $\mathcal{W}_2$ remains constant for each cyclic shift of the edge sequence representing the double Eulerian cycle. In Figure 12, $A = (e_5, e_1, e_2, e_2, e_3, e_3, e_1, e_5, e_4, e_4)$ is a double Eulerian walk. As before, visitation times are in parentheses. The distance between the first appearance at each edge e_1, e_2, e_3, e_4, e_5 and the second appearance in walk A are 5,1,1,1,3, respectively. Walk $B = (e_1, e_2, e_2, e_3, e_3, e_1, e_5, e_4, e_4, e_5)$ is the sequence generated by cyclically shifting walk A by one. The distances between the two occurrences of each edge in walk B are unchanged.

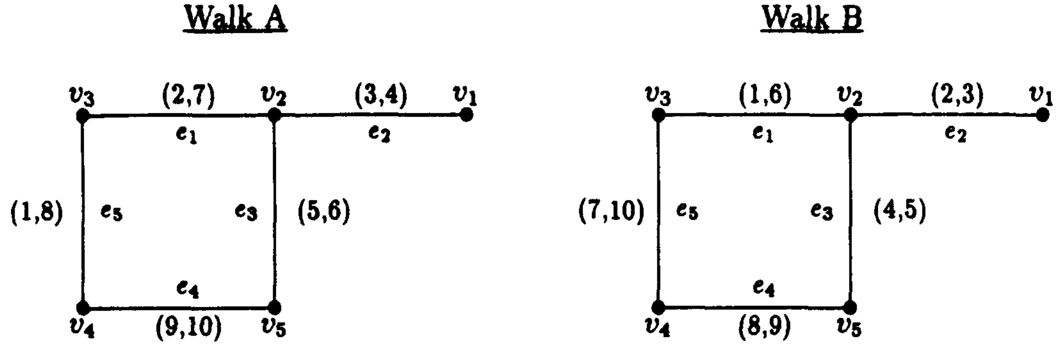


Figure 12. *Double Eulerian cycles with visitation times (cycled)*

In a graph G with size q, the sequence of edges $f_1, f_2, \ldots, f_{2q}$ in a double Eulerian cycle completely determines the measure of the walk. More precisely, the measure of a walk is ultimately determined by the number of edges previously traversed exactly once (called *single-edges or σ-edges*) along the double Eulerian cycle. When an edge e has been traversed twice (or not at all) during the walk, subsequent edges along the walk can neither increase nor decrease the difference of the two visitation times for that edge. *When an edge e', however, has been traversed exactly once during the walk, each subsequent edge traversed along the walk increases the difference of the visitation time for e' by one.* Thus, it is the number of current σ-edges at each step along the walk that determines the measure of the double Eulerian cycle. Finally, as stated before, the aggregate sum of the σ-edges at each vertex along the double Eulerian cycle, is the measure of the walk. From this perspective, the measure of a double Eulerian cycle can be calculated as follows:

1. Transform the sequence of edges $\{e_n\}_{n=1}^{2q}$ in a double Eulerian cycle into a sequence, $\{a_n\}_{n=1}^{2q}$ of 1's and -1's as follows:

$$\text{Edge } \{x,y\} = \begin{cases} 1 & \text{if this is the first occurrence of the edge } \{x,y\} \\ & \text{on the double Eulerian cycle} \\ -1 & \text{otherwise.} \end{cases}$$

2. Let $\{t_n\}_{n=1}^{2q}$ be the sequence of partial sums of the series $\sum_{n=1}^{2q} a_n$.

3. The measure of the walk is $\sum_{k=1}^{2q} t_j$.

For example, consider walk A in Figure 11. We find the following:

1. $\{a_n\}_{n=1}^{10} = \{1,1,1,1,1,-1,-1,-1,-1,-1\}$.

2. $\{t_n\}_{n=1}^{10} = \{1,2,3,4,5,4,3,2,1,0\}$.

3. The measure of walk A is $\sum_{k=1}^{2q} t_j = 25$.

It is interesting to note that set of possible sequences $\{a_n\}_{n=1}^{2q}$ of 1's and -1's for a graph G defined in this way is a subset of all of the (north/east) routes between opposite corners on a $q \times q$ lattice that are on or below the diagonal, where q is the size of G. It is well-known [Ref. 22] that the total number of subdiagonal routes is

$$2C_n = \frac{2}{n+1}\binom{2n}{n},$$

where C_n is the nth Catalan number.

In the remainder of this section we show that the value of a graph G can be determined as a function of the edge-status, es_v, of a vertex v where $v \in EM(G)$ and q, the size of G. First we develop some results relating the concepts of status and edge-status.

Theorem IV.3 *Let $T = (V, E)$ be a tree. Then $es_v = s_v - q$ where es_v is the edge-status of v, s_v is the status of v, and q is the size of T.*

Proof: If a tree T has one edge, it follows immediately that $es_v = s_v - q$. Assume T has more than one edge. Let v be a root of T that gives T a height of h. Assume e_1 and e_2 are two edges such that $d_e(v, e_1) = d_e(v, e_2) = k$ and where e_1 and e_2 are incident with the same vertex v_3 with $d(v, v_3) = k+1$. Then $(v, \ldots, e_1, v_3)$ and $(v, \ldots, e_2, v_3)$ are two different walks from v to the vertex v_3. This infers a cycle in T. Therefore, each edge $e \in E$, where $d_e(v, e) = i$, $0 \leq i \leq h-1$ is incident with a *unique* vertex $n \in V - v$ where $d(v, n) = i + 1$. Since $|E| = |V| - 1$, it follows that

$$es_v = \sum_{e \in E} d_e(v, e) = \sum_{n \in V - v} d(v, e) - 1 = s_v - q.$$

■

The following corollary follows immediately from Theorem IV.3.

Corollary IV.4 *Let T be a tree, with edge-median $EM(T)$ and median $M(T)$. Then $EM(T) = M(T)$.*

Unfortunately, if a graph G is not a tree, it is not necessarily true that $EM(G) = M(G)$. Graph G in Figure 13 has $EM(G) = \{v_6\}$ and $M(G) = \{v_5\}$.

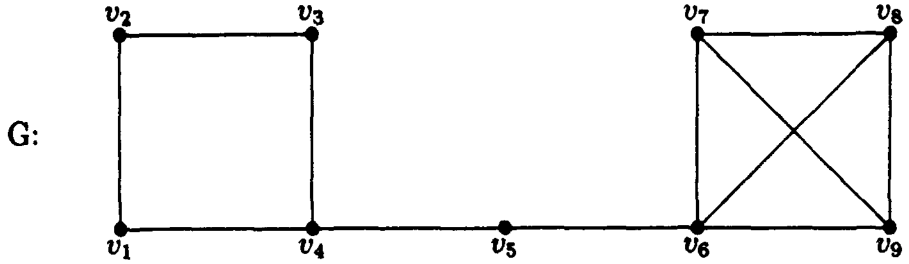


Figure 13. *Example for which $EM(G) \neq M(G)$*

The next result provides a lower bound for the measure of a walk in W_{2_v} where $v \in V$ on a connected graph $G = (V, E)$.

Lemma IV.5 *Let $G = (V, E)$ be a connected graph of size q. Let $v \in V$, with edge-status es_v. If m is the minimum measure of a double Eulerian cycle W_{2_v} beginning at v, then $m \geq 2(es_v) + q$.*

Proof: Let v be the initial vertex of a double Eulerian cycle in G. Let $\max\limits_{e \in E} d_e(v, e) = n$. Since a double Eulerian cycle is closed, all of the members of the set of edges of edge-distance j, $1 \leq j \leq n$, from v must be visited twice before all edges of edge-distance $j - 1$ are visited twice. It follows that there are at least j σ-edges when any edge e_j in the set of edges of distance j from v is visited for the first time in a double Eulerian walk. Furthermore, there are at least $j + 1$ σ-edges when e_j is traversed the second time in the walk (since e_j is now included in the set of σ-edges). Therefore, each edge e_j at edge-distance j from v increases the measure of a walk by at least $2j+1$. It follows that if m is the minimum measure of a double Eulerian walk beginning at v, then

$$m \geq \sum_{e \in E} (2\,(d_e(v,e)) + 1) = 2(es_v) + q.$$

■

We now describe an algorithm (Algorithm A) to construct a double Eulerian cycle of minimum measure on a connected graph. The algorithm includes a subroutine **Cycle1** that is recursively called throughout the algorithm. Informally, Algorithm A is very similar to the depth-first search algorithm [Ref. 23]. The algorithm proceeds from the initial vertex v in a *forward direction* (adding new edges) for as long as this is possible. When it is no longer possible to advance (add a new edge), the algorithm backtracks to the first vertex from which it is then possible to go *forward* revisiting edges as it goes. The algorithm proceeds until each edge is visited exactly two times. Each new edge added by the algorithm has the following properties:

1. The new edge is not in the current walk,
2. The new edge is incident with the last visited vertex in the walk, and
3. The edge-distance from v to the new edge is exactly one greater than the edge-distance from v to the last visited σ-edge in the walk.

It follows from the above that when no new edge can be added to the walk, the next edge traversed is the last visited σ-edge in the walk (i.e., the σ-edge of greatest edge-distance from the initial vertex v). This is in essence the backtracking aspect of the algorithm.

ALGORITHM A [Constructs a double Eulerian cycle on a connected graph $G = (V, E)$ beginning at vertex $\nu \in V$]

Input: A graph $G = (V, E)$ of size q and a vertex ν as global parameters.

Output: Array P, an ordered list of edges constituting a double Eulerian cycle through G beginning at the given vertex ν.

Parameters $N = [n_1, \ldots, n_{2q+1}]$, $P = [p_1, \ldots, p_{2q}]$, and $H = [h_1, \ldots, h_q]$, are global arrays of vertices, edges, and edges of size $2q + 1$, $2q$, and q, respectively.

(Note: The parameter d is an integer that denotes the edge-distance of the last visited σ-edge from vertex ν in an ongoing walk. When there are no σ-edges in the walk, d is set to -1. The parameter n_i indicates the current vertex in the walk. The parameters i and j are integer counters indicating the number of edges currently traversed and the number of current σ-edges along the walk, respectively.)

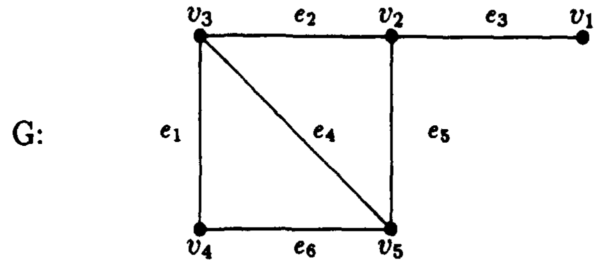


Figure 14. *A graph to illustrate Algorithm A*

Algorithm *A*

Initially set $j := 1$; $i := 1$; $d := -1$; $n_1 := \nu$; $n_k :=$ empty; $2 \leq k \leq 2q + 1$; $p_l :=$ empty, $1 \leq l \leq 2q$; and $h_m :=$ empty; $1 \leq m \leq q$.

Cycle1(N, P, H, j, i, d); **end.**

Subroutine Cycle1 (N, P, H, j, i, d)

```
if $d_e(n_i, e) = 0$ and $d_e(n_1, e) = d + 1$ and $d_e(\nu, e) = d(\nu, n_i)$ and $e \notin P$, for $e \in E$
  then $n_{i+1} := x$ where $e = (n_i, x)$
       $p_i := e$
       $h_j := e$
       $d := d + 1$
       $i := i + 1$
       $j := j + 1$
       Cycle1$(N, P, H, j, i, d)$
  else $p_i := h_j$
       $n_{i+1} := n_{i-1}$
       $d := d - 1$
       $i := i + 1$
       $j := j - 1$
       if $i = 2q$
         then end.
         else Cycle1$(N, P, H, j, i, d)$
```

To illustrate Algorithm A, consider the graph G in Figure 14. Let v_4 be the initial vertex ν. Both e_1 and e_6 satisfy the criteria for being a new edge to add to the walk. Without loss of generality select e_1. Subsequently, both e_2 and e_4 satisfy the criteria for a new edge to enter the walk. Let e_4 be selected. Continuing, the double Eulerian cycle constructed by Algorithm A is the sequence of edges $P = [e_1, e_4, e_4, e_2, e_3, e_3, e_2, e_1, e_6, e_5, e_5, e_6]$. Alternatively, if e_2 had been selected rather than e_4, the sequence of edges $P' = [e_1, e_2, e_3, e_3, e_2, e_4, e_4, e_1, e_6, e_5, e_5, e_6]$ is generated. Both F and F' have the same measure.

The algorithm adds a new edge to the walk when the three conditions stated above are satisfied. Each *new* edge has an edge-distance from ν of exactly one greater than the last σ-edge visited. During the walk if a new edge cannot be traversed, the algorithm backtracks through the last visited σ-edge until the criteria for adding a new edge is met. When a σ-edge is added through backtracking, the σ-edge to be traversed is the one that has the greatest edge-distance from ν of all σ-edges currently in the walk. This is clearly the last visited σ-edge. The following lemma shows that ultimately each edge in the graph is traversed exactly twice.

Lemma IV.6 *Let $G = (V, E)$ be a connected graph. Algorithm A constructs a double Eulerian cycle on G beginning at $v \in V$.*

Proof: We prove, by induction on the edge-distance from the initial vertex v, that each edge is traversed exactly twice.

Basis Step: The edge-distance between the first edge e on the walk generated by Algorithm A and the initial vertex v is 0. Since there are only a finite number of edges, the edge-distance from v to each edge is finite. Therefore, the walk must eventually backtrack through each of the σ-edges of the ongoing walk generated by the algorithm. When there is no edge of distance 1 to be added after edge e, tnen edge e will be revisited by backtracking. Hence e is traversed exactly 2 times. In a similar fashion when the walk continues along another edge e' of edge-distance 0 from v, finiteness and backtracking insure that e' is traversed exactly twice.

Inductive Step: Assume that the algorithm traverses each edge e where $d_e(v,e) = k$. Let e_{k+1} be an edge of edge-distance $k+1$ from v. There exists an edge e_k of edge-distance k incident with edge e_{k+1}, else e_{k+1} is not of distance $k+1$ from v. Let vertex v_k be incident to both e_k and e_{k+1}. The distance from v to v_k is k, else either the edge-distance to e_k is not k or the edge-distance to e_{k+1} is not $k+1$. When e_k is traversed for the first time, the edge e_{k+1} satisfies the criteria to enter the walk. Either e_{k+1} or another edge e'_{k+1} of edge distance $k+1$ will be the next edge added to the walk. If e_{k+1} is not the next edge in the walk, then because there are only a finite number of edges and by backtracking, the walk will eventually return to v_k. The edge e_{k+1} still satisfies the criteria to enter the walk. Since there are only a finite number of edges incident to v_k, e_{k+1} must ultimately enter the walk and be traversed exactly twice through backtracking. ■

We made the point that when an edge of distance k from ν is added to a walk in Algorithm A, it is added when an edge of distance $k-1$ is the last σ-edge in the walk. Now we make a further statement about the set of σ-edges when an edge of distance k is added to a walk.

Lemma IV.7 *When an edge e_k of edge-distance k from the initial vertex of the walk is traversed for the first time using Algorithm A, there are exactly k edges that have been previously traversed exactly once.*

Proof: We prove, by induction on the edge-distance, that when an edge e_k of edge-distance k from the initial vertex v of the walk is traversed for the first time using Algorithm A, there are exactly k σ-edges on the walk.

Basis Step: When an edge e of edge-distance 0 is traversed for the first time, we are at the vertex v and there are no σ-edges remaining in the walk.

Inductive Step: Assume that when any edge of edge-distance k is traversed using Algorithm A for the first time there are exactly k σ-edges in the ongoing walk. Let $d_e(v, e_{k+1}) = k+1$. The edge e_{k+1} is initially traversed only when the last visited σ-edge ε in the walk has an edge-distance of k from v. Therefore, by the inductive hypothesis, there are then exactly $k+1$ σ-edges after e_{k+1} initially enters the walk. ■

Now we can relate the measure of double Eulerian cycles in a graph G to the edge-status of G.

Theorem IV.8 *Let $G = (V, E)$ be a connected graph of size q. Let $v \in V$, with edge-status es_v. If m is the minimum measure of a double Eulerian cycle W_v beginning at v, then $m = 2(es_v) + q$.*

Proof: Algorithm A constructs a double Eulerian cycle through a graph. When an edge e_j of edge-distance j from the initial vertex in the walk is traversed for the first time using Algorithm A, Lemma IV.7 states there are exactly j σ-edges. This then adds j to the measure of the walk (1 for each of the j σ-edges). When an edge e_j of edge-distance j is traversed by Algorithm A for the second time, there are exactly $j+1$ σ-edges (the edge e_j now belongs to the set of σ-edges). This adds $j+1$ to the measure. Therefore, the measure of a double Eulerian cycle constructed by Algorithm A is

$$\sum_{e \in E} (d_e(v,e) + d_e(v,e) + 1) = \sum_{e \in E} (2d_e(v,e) + 1) = 2(es_v) + q.$$

It follows immediately from Lemma IV.5 that the minimum measure of a double Eulerian cycle W_{2_v} beginning at vertex v is $2(es_v) + q$. ■

Corollary IV.9 *Let G be a connected graph. Let W_{2_v} be a double Eulerian cycle on G that produces the value of G. Then $v \in EM(G)$.*

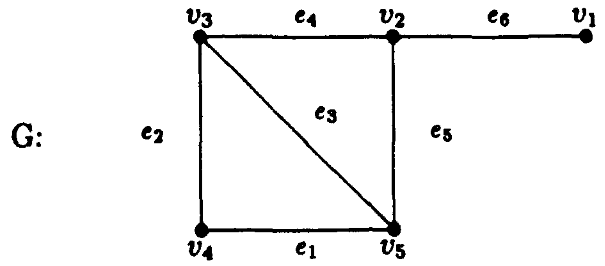


Figure 15. *To illustrate the value of a graph*

Proof: Follows directly from Theorem IV.8 and the definition of $EM(G)$. ■

Corollary IV.10 *Let $G = (V, E)$ be a connected graph of size q with a value of m. Let $v \in EM(G)$, with edge-status es_v. Then $m = 2(es_v) + q$.*

Proof: Follows immediately from Theorem IV.8. ■

Corollary IV.11 *Let $T = (V, E)$ be a tree of size q. Let $v \in V$, with edge-status es_v. The measure m of a double Eulerian cycle W_v in T is $m = 2(s_v) - q$.*

Proof: Follows directly from Theorem [illegible] and Theorem IV.8. ■

We have shown that the value of a connected graph G is determined by the edge-status, es_v of a vertex $v \in EM(G)$ and by q the number of edges in G. Algorithms to find the median of a graph can be found in [Ref. 24] and [Ref. 25]. These algorithms can be modified in a straightforward manner to determine the distances from vertices to edges and the edge-median of a graph.

We discussed earlier that the measure of a walk $W \in \mathcal{W}_2$ can be defined using the distance between the two occurrences of each edge on the cycle W. If each cyclic shift of a cycle is considered equivalent, this definition assigns the same measure to equivalent cycles. In Figure 15, the double Eulerian cycle,

$$W = (e_1, e_5, e_4, e_3, e_3, e_2, e_2, e_4, e_6, e_6, e_5, e_1),$$

and each cyclic shift of W has the measure 12 when using the distance between identical edges to calculate the measure.

C. THE VALUE OF A DIRECTED GRAPH

The following well-known theorem is stated without proof. See, e.g., [Ref. 19].

Theorem IV.12 *A weak digraph is Eulerian if and only if for every vertex its in-degree and out-degree are equal.*

From this theorem, the next result follows readily.

Lemma IV.13 *Every double Eulerian walk in a digraph D is closed.*

Proof: Let $P = (u = u_0, u_1, \ldots, u_{2k} = v)$ be a double Eulerian cycle in the digraph $D = (V, A)$ where $|A| = k$. If $u \neq v$, then v appears an even number of times on P, since each incoming arc on v is used exactly twice. Since v is the terminal vertex of P, however, v has been exited an odd number of times.

This is a contradiction, since P is double Eulerian. Thus $u = v$, and P is a closed walk. As an immediate corollary, $\text{in}(u) = \text{out}(u)$, and D is Eulerian. ■

The following theorem shows that the Eulerian and double Eulerian property of a digraph are equivalent.

Theorem IV.14 *A weak digraph D is double Eulerian if and only if D is Eulerian.*

Proof: Suppose that a digraph D is double Eulerian. Since a double Eulerian cycle provides a path between any pair of vertices, the digraph must be strong (and hence also weak). By Lemma IV.13, the in-degree and out-degree of each vertex is even. Therefore, D is Eulerian. Conversely, suppose D is Eulerian. Traversing an Eulerian trail exactly twice yields a double Eulerian cycle. ■

The Eulerian graph can be partitioned into edge disjoint cycles.

Theorem IV.15 *A digraph D has an edge-factor if and only if it is Eulerian.*

Proof: Let D be an Eulerian digraph. A closed Eulerian trail on D is an edge-factor. Conversely, let F be an edge-factor of the digraph D. The in-degree and out-degree of any vertex in each cycle of F are equal. Since the arcs on the cycles in F partition the set of arcs in D, the in-degree and out-degree of each vertex in D must be equal. Therefore, D is Eulerian. ■

By Theorem IV.15 every double Eulerian cycle W_{2_v} along a digraph D defines a specific edge-factor F. Given F we can obtain a lower bound for the measure of W_{2_v}.

Theorem IV.16 *Let $F = \{C_1, C_2, \ldots, C_f\}$ be an edge-factor of the digraph D. The measure, m, of any double Eulerian cycle W_{2_v} in D satisfies*

$$m \geq \sum_{i=1}^{f} |C_i|^2 + \rho_v(F).$$

Proof: We prove this by using strong induction on k, where k is the largest number of cycles in any edge-factor of D.

Basis Step: Let D be a digraph such that for any edge-factor F of D, $|F| = 1$ (i.e., no edge-factor of D has more than a single cycle). Therefore the set E of cycles in each edge-factor of D is the set of Eulerian cycles in D. Then $F = \{R\}$ where $R \in E$. Every double Eulerian cycle in $\mathcal{W}_{2_v}$ along D is exactly two traversals of such a cycle R. Therefore, the measure of any walk in $\mathcal{W}_{2_v}$ is

$$|R|^2 = \sum_{C \in F} |C|^2 + \rho_v(F).$$

(Note: $\rho_v(F) = 0$ since all vertices v are on R and $C = R$ is the only cycle in F.)

Inductive Step: Let $F = \{C_1, C_2, \ldots, C_f\}$ be an edge-factor of the digraph D. Assume that the value m of any double Eulerian cycle W_{2_v} in D satisfies

$$m \geq \sum_{i=1}^{f} |C_i|^2 + \rho_v(F).$$

Let D be a digraph satisfying $\max_{F}|F| = n$, and let W_{2_v} be a double Eulerian cycle on D. Since D is finite, then at some point along W_{2_v} a vertex is repeated. This defines a cycle in D. Let C be the first cycle completed along W_{2_v}. Let α be the k^{th} arc traversed on C. After α is initially traversed there are exactly k σ-arcs in C. Let α' be the j^{th} arc on C to be traversed for the second time along W_{2_v}. After α' is traversed for the second time, there are $|C| - j$ σ-arcs in C. Furthermore, each step along C increases the difference of the visitation time by one for each σ-arc not in C. Since each arc in C is traversed twice, C increases by 2 the difference of the visitation times for exactly one arc of each edge-distance i from v (by the definition of C), where $0 \leq i \leq d_e(v,C) - 1$. Therefore, the set of arcs in C taken together increases the measure of the walk W_{2_v} by some

$$\varepsilon_C \geq \sum_{k=1}^{|C|} k + \sum_{j=1}^{|C|}(|C| - j) + 2|C|d_e(v,C) = |C|^2 + 2|C|d_e(v,C).$$

Let $D' = D - (C)_v$. After we remove C, the edge-factor for each *component* of D' has at most $n-1$ cycles. Let $F' = \{C_1', C_2', \ldots, C_f'\}$ be an edge-factor of the digraph D'. By the inductive hypothesis the measure m of any double Eulerian cycle on any component G of D' satisfies

$$m \geq \sum_{i=1}^{f} |C_i'|^2 + \rho_v(F').$$

Therefore, if $F = \{C_1, C_2, \ldots, C_f\}$ is an edge-factor of the digraph D, then the measure m for any double Eulerian cycle W_{2_v} along D satisfies

$$m \geq \sum_{i=1}^{f} |C_i|^2 + \rho_v(F).$$

■

We now describe an algorithm (Algorithm D) that constructs a double Eulerian cycle of minimum measure, beginning at a vertex v on an Eulerian digraph. Algorithm D adds cycles to create a double Eulerian cycle whose measure attains the lower bound of Theorem IV.16. This algorithm is similar to Algorithm A. The reader should make note of the similarities and differences between the roles played by the edges of a graph in Algorithm A and the cycles in the Eulerian digraph in Algorithm D. Informally speaking, Algorithm D moves *forward*, entering cycles of greater distance from a vertex v for as long as this is possible. When it is no longer possible to enter a new cycle at greater distance, the algorithm traverses each arc on the current cycle exactly twice and then backtracks to the previous cycle. The backtracking continues to the first cycle from which it is possible to go forward to enter new cycles. It is clear that a cycle C may be entered at several places along a given walk. The first edge traversed on C cannot be at distance $d > d_e(v, C)$. Algorithm D does not allow a cycle to be entered for the first time at any vertex other than a vertex of distance $d_e(v, C)$.

Algorithm D includes a subroutine **Cycle2** that is recursively called throughout the algorithm. Algorithm D constructs a double Eulerian cycle W_{2_v} by successively selecting cycles from an edge-factor to be traversed. Each new cycle C added has the following properties:

1. There are $d_e(v, C)$ arcs in W_v that have been traversed exactly once when C is entered.
2. No arc on the cycle C has previously been traversed.
3. If n is the last vertex encountered along the walk, then $n \in (C)_v$ and $d(v, n) = d_e(v, C)$.

ALGORITHM D [Constructs a double Eulerian cycle on an Eulerian digraph beginning at vertex ν]

Input: An Eulerian digraph $D = (V, A)$ of size q, a vertex $\nu \in V$, and edge-factor $C = \{c_1, c_2, \ldots, c_f\}$ as global parameters.

Output: Array P, an ordered list of the arcs constituting a double Eulerian cycle through D beginning at the given vertex ν.

Parameters: $N = [n_1, \ldots, n_{2q+1}]$, $P = [p_1, \ldots, p_{2q}]$, and $M = [m_1, \ldots, m_f]$, are global arrays of vertices, arcs, and integers of size $2q + 1$, $2q$, and f, respectively. Parameters i, j, and t are integers.

The following is a brief description of specific parameters:

1. The parameter n_i indicates the current vertex in the walk.
2. The value of m_l indicates the following about the arcs in the cycle c_l:
   (a) If $m_l = 0$ then no arcs on c_l are currently on the walk,
   (b) If $m_l = 1$ then at least one, but not every arc in c_l is currently on the walk,
   (c) If $m_l = 2$ every arc in c_l has been traversed at least once.
3. The parameter t indicates the number of σ-arcs currently in the walk.

Algorithm D

Initially set $j := 0$; $i := 1$; $t := 0$; $n_1 := \nu$; $n_k :=$ empty, $2 \leq k \leq 2q+1$; $p_s :=$ empty, $1 \leq s \leq 2q$; and $m_h := 0$, $1 \leq h \leq f$.

Cycle2(N, P, M, C, j, i, t); **end.**

Subroutine Cycle2(N, P, M, C, j, i, t)

```
(D1) if $[[d(\nu, n_i) = d_e(\nu, c_l) = t]$ and $[\mathrm{m}_l = 0]]$ where $n_i \in c_l, l \in \{1, 2, \ldots, f\}$
        then $j := j + 1$
             $Temp := c_l$
             $c_l := c_j$
             $c_j := Temp$
             $Temp := m_l := m_l + 1$
             $m_l := m_j$
             $m_j := Temp$
             $n_{i+1} := x$ where $(n_i, x) \in c_j$
             $p_i := (n_i, n_{i+1})$
             $i := i + 1$
             $t := t + 1$
             Cycle2$(N, P, M, C, j, i, t)$
(D2)    else if $[[d(\nu, n_i) = d_e(\nu, c_j) = (t - |c_j|)]$ and $[\mathrm{m}_j = 1]]$ where $n_i \in c_j$
             then $n_{i+1} := x$ where $(n_i, x) \in c_j$
                  $p_i := (n_i, n_{i+1})$
                  $i := i + 1$
                  $\mathrm{m}_j := \mathrm{m}_j + 1$
                  $t := t - 1$
                  Cycle2$(N, P, M, C, j, i, t)$
(D3)         else if $[[d(\nu, n_i) = d_e(\nu, c_j) = t]$ and $[m_j = 2]]$ where $n_i \in c_j$
                  then $j := j - 1$
                       Cycle2$(N, P, M, C, j, i, t)$
(D4)              else if $m_j = 1$
                       then $n_{i+1} := x$ where $(n_i, x) \in c_j$
                            $p_i := (n_i, n_{i+1})$
                            $i := i + 1$
                            $t := t + 1$
                            Cycle2$(N, P, M, C, j, i, t)$
(D5)                   else if $m_j = 2$
                            then $n_{i+1} := x$ where $(n_i, x) \in c_j$
                                 $p_i := (n_i, x)$
                                 $i := i + 1$
                                 $t := t - 1$
                                 Cycle2$(N, P, M, C, j, i, t)$
(D6)                        else end
```

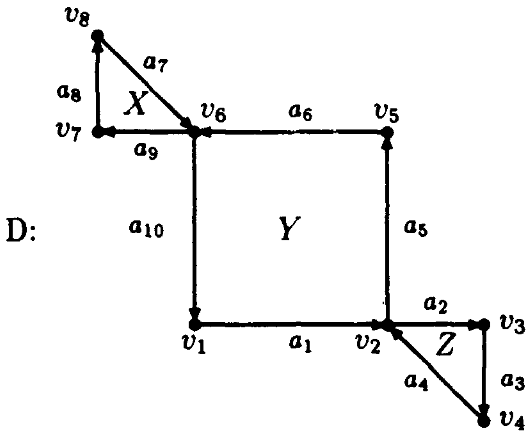


Figure 16. *A digraph to illustrate Algorithm D*

The following is a brief description of the conditional lines in Algorithm D:

1. Line (D1) insures that:

   (a) There are $d_e(\nu, c_l)$ σ-arcs in the walk when c_l is initially entered since $d_e(\nu, c_l) = t$,

   (b) No arc on the cycle c_l has previously been traversed ($m_l = 0$),

   (c) The distance to the last vertex in the walk equals the edge-distance to c_l from the initial vertex ν, i.e., $d(\nu, n_i) = d_e(\nu, c_l)$.

   (The current cycle goes from c_j to c_{j+1}.)

2. Line (D2) insures that at the completion of *one* traversal of the cycle c_j another cycle is not entered for the first time, rather, the walk continues to traverse c_j for a second time.

3. Line (D3) insures that when every arc in a cycle c_j is traversed exactly twice, the walk continues along the cycle c_{j-1}.

4. Lines (D4) and (D5) guide the walk along the current cycle c_j unless it is time to initially enter the cycle c_{j+1} (D1), to reenter the cycle c_{j-1} (D3), or to begin the second traversal around the cycle c_j (D2). Line (D4) is satisfied during the first traversal of c_j while (D5) is satisfied during the second traversal of c_j.

5. Line (D6) ends the procedure.

To .llustrate Algorithm D, consider the digraph D in Figure 16. Let C be the edge-factor consisting of the cycles $X = (a_7, a_9, a_8)$, $Y = (a_1, a_5, a_6, a_{10})$, and $Z = (a_2, a_3, a_4)$. Initially set $n_1 = v_1$; $p_s :=$ empty, $1 \leq s \leq 20$; $M = [0,0,0]$; $C = [X, Y, Z]$; $j := 0$; $i := 1$; $t := 0$;

The conditional lines in Algorithm D are satisfied in the following order:

- **Cycle2**$(N, P, M, C, 0, 1, 0)$ is called and the conditions for line D1 are satisfied with $c_l = c_2 = Y$. The values for the parameters are currently $j = 1$, $C = [Y, X, Z]$, $M = [1, 0, 0, \ldots, 0]$, $n_2 = v_2$, $p_1 = a_1$, $i = 2$, and $T = 1$.
- **Cycle2**$(N, P, M, C, 1, 2, 1)$ is called and the conditions for line D1 are satisfied with $c_l = c_3 = Z$. The values for the parameters are currently $j = 2$, $C = [X, Z, Y]$, $M = [1, 1, 0]$, $n_3 = v_3$, $p_2 = a_2$, $i = 2$, and $T = 2$.
- **Cycle2**$(N, P, M, C, 2, 2, 2)$ is called and the conditions for line D4 are satisfied. The values for the parameters are currently $j = 2$, $C = [X, Z, Y]$, $M = [1, 1. 0]$, $n_4 = v_4$, $p_3 = a_3$, $i = 3$, and $T = 3$.
- **Cycle2**$(N, P, M, C, 2, 3, 3)$ is called and the conditions for line D4 are satisfied. The values for the parameters are currently $j = 2$, $C = [X, Z, Y]$, $M = [1, 1, 0]$, $n_5 = v_2$, $p_4 = a_4$, $i = 4$, and $T = 4$.
- **Cycle2**$(N, P, M, C, 2, 4, 4)$ is called and the conditions for line D2 are satisfied with $c_j = c_2 = Y$. The values for the parameters are currently $j = 2$, $C = [X, Z, Y]$, $M = [1, 2, 0]$, $n_6 = v_3$, $p_5 = a_2$, $i = 4$, and $T = 3$.
- **Cycle2**$(N, P, M, C, 2, 4, 3)$ is called and the conditions for D5 are satisfied. The values for the parameters are currently $j = 2$, $C = [X, Z, Y]$, $M = [1, 2, 0]$, $n_7 = v_4$, $p_6 = a_3$, $i = 4$, and $T = 2$.
- The remaining conditional lines are satisfied as follows: D5, D3, D4, D4, D4, D2, D5, D5, D1, D4, D4, D2, D5, D5, D3, D4, and D6 ends the program.

We find the sequence of arcs traversed to be

$$P = [a_1, a_2, a_3, a_4, a_2, a_3, a_4, a_5, a_6, a_{10}, a_1, a_5, a_6, a_9, a_8, a_7, a_9, a_8, a_7, a_{10}].$$

We now show that Algorithm D constructs a double Eulerian cycle W whose measure achieves the lower bound given in Theorem IV.16. First we need to show a relationship between adjacent cycles in an edge-factor of a digraph.

Lemma IV.17 *For an edge-factor F of $D = (V, A)$, let $C \in F$ and v a vertex where $d_e(v, C) > 0$. There exists a cycle $P \in F$ adjacent to C such that $d_e(v, P) < d_e(v, C)$.*

Proof: Let F be an edge-factor of D where $C, P \in F$. Let $d(v, x) = d_e(v, C)$ where $x \in (C)_v$ and $\{\alpha_1, \ldots, \alpha_j\}$ is the set of arcs incident to x. Without loss of generality, let α_1 and α_2 be arcs in C. Then the minimal length walk W between v and x does not include either of α_1 and α_2 else there is a vertex $z \in (C)_v$ such that $d(v, z) < d(v, x)$. Without loss of gererality, let α_k be the arc incident to x in W. Therefore, C is adjacent to a cycle P where $x \in (P)_v$ and $d_e(v, P) < d_e(v, C)$. ■

We now show that the walk constructed by Algorithm D visits every edge exactly twice.

Lemma IV.18 *For a given edge-factor $F = \{C_1, C_2, \ldots, C_f\}$ of the digraph $D = (V, A)$ and a vertex $v \in V$, Algorithm D constructs a double Eulerian cycle along D beginning at vertex v.*

Proof: We prove, by strong induction on the edge-distance to a cycle $C \in F$ from the initial vertex v, that each arc in D is traversed exactly twice.

Basis Step: Let F be an edge-factor of a digraph D. The initial step of the walk beginning at vertex v is along a cycle C_0 where $d_e(v, C_0) = 0$. The walk continues along the arcs of C_0 until each of the 3 criteria (on page 64) are satisfied for a new cycle to enter the walk. The edge-distance from v to a new cycle added to W_v is greater than the distance of the current cycle. As there are only a finite number of arcs in each cycle and only a finite number of cycles, eventually during the traversal of any cycle there will be no adjacent cycles of greater edge-distance that have not been initially entered. When the current cycle is exited it is to a cycle of smaller edge-distance from v. This can only occur after each of the arcs of the current cycle have been traversed exactly twice. Since there are only a finite number of cycles and C_0 is the cycle of smallest distance on the walk, C_0 must eventually be revisited and traversed exactly twice. A similar argument insures that every cycle of distance 0 is traversed exactly twice.

Inductive Step: We now assume that the algorithm traverses each cycle C_n of distance n from v exactly twice for all $n < k$. Let C_k be a cycle of edge-distance k from v. By Lemma IV.17 there exists an arc α_{k-1} of edge-distance $k-1$ on a cycle C_i adjacent to C_k where $d_e(v, C_i) = i$, $0 \leq i < k$. Therefore, on the first or second traversal of C_i, the conditions for the new cycle C_k to enter the walk are satisfied and C_k is entered. From the discussion above, C_k will then be traversed exactly twice. ■

We have seen that Algorithm D completes a double Eulerian cycle. We now discuss the number of σ-arcs existant at each step of the walk constructed by Algorithm D. The following lemma shows that we can obtain the lower bound of Theorem IV.16.

Lemma IV.19 *Let F be an edge-factor of the digraph $D = (V, A)$. The minimum measure of a cycle W_{2_v} generated by Algorithm D is $\sum_{C \in F} |C|^2 + \rho_v(F)$, where $\rho_v(F)$ is the posture for the factor F.*

Proof: By Lemma IV.18, Algorithm D produces a walk W_{2_v} that traverses each arc of an Eulerian digraph exactly 2 times. Furthermore, when the first arc, α_1, in a cycle C_j of distance j from v is traversed in Algorithm D for the first time, there are exactly j σ-arcs along W_{2_v}. This follows from line (D1) of Algorithm D, the conditional statement governing when new cycles can initially enter W_{2_v}. When the k^{th} arc α_k in cycle C_j is traversed for the first time, there are exactly $j+k-1$ σ-arcs in W_v. Therefore, the first traversal of each arc in C_j adds $\sum_{k=1}^{|C_j|}(j+k-1)$ to the measure of the walk. When the k^{th} arc of the cycle C_j is traversed for the second time, there are $j + |C| + 1 - k$ σ-arcs in W_{2_v}. Therefore, the second traversal of each of the arcs in C_j adds

$\sum_{k=1}^{|C_j|}(j + |C| + 1 - k)$ to the measure of W_{2_v}. It follows then that including the cycle C_j in the walk W_{2_v} contributes

$$\left(\sum_{k=1}^{|C|}(j + k - 1)\right) + \left(\sum_{k=1}^{|C|}(j + |C| + 1 - k)\right)$$

$$= \frac{(j + |C| - 1)(j + |C|)}{2} - \frac{(j-1)(j)}{2} + \frac{(j + |C|)(j + |C| + 1)}{2} - \frac{(j)(j+1)}{2}$$

$= |C|^2 + 2j|C|$ to the measure of W_{2_v}. Summing over all cycles in F yields

$$\sum_{C \in F}(|C|^2 + 2j|C|) = \sum_{C \in F}(|C|^2 + 2d_e(v, C)|C|) = \sum_{C \in F}|C|^2 + \rho_v(F).$$

Hence, the minimum measure of W_{2_v} is $\min_F \sum_{C \in F} |C|^2 + \rho_v(F)$. ■

Theorem IV.20 *Let F be an edge-factor of the digraph $D = (V, A)$. The value of D is $\min_F \sum_{C \in F} |C|^2 + \rho_v(F)$, where $\rho_v(F)$ is the posture for the factor F.*

Proof: Follows directly from Lemma IV.19 and Lemma IV.16. ■

Corollary IV.21 *Let D be an Eulerian digraph. Let W_v be a double Eulerian cycle on D that produces the value of the graph. Then $v \in M_F(D)$.*

Proof: Follows immediately from Theorem IV.20 and the definition of $M_F(D)$, the mean of the factor F. ■

		Size of the Cycle						
		1	2	3	4	5	6	7
Distance from v	0	1	2	3	4	5	6	7
	1	3	4	5	6	7	8	9
	2	5	6	7	8	9	10	11
	3	7	8	9	10	11	12	13
	4	9	10	11	12	13	14	15

Table V. *Average cost of arcs in particular cycles*

We have shown that the value, ν, of an Eulerian digraph D is given by

$$\nu = \min_{F} \sum_{C \in F} |C|^2 + \rho_v(F), \tag{IV.1}$$

where F is an edge-factor of D.

Table V lists the average amount each arc α adds to the measure of the walk when α is in a cycle of size $|C|$ for $1 \leq |C| \leq 7$ where $0 \leq d_e(v, C) \leq 4$ from the initial vertex v. It follows from the discussion above that the value in the (i, j) position in Table V is $2i + j$.

We can modify algorithms that find the median of a graph in a straightforward manner to find the posture of a directed graph for a given edge-factor. Finding all the edge-factors of a directed graph, however, can be extremely difficult. It is not obvious how to determine *a priori* an edge-factor that produces the double Eulerian cycle of minimum measure, hence the value, of a digraph. The following two lemmas, however, remove certain edge-factors of a digraph D from consideration when generating a cycle with minimum measure.

Lemma IV.22 *Let F be a reducible edge-factor of $D = (V, A)$ where $X \in F$ can be reduced to cycles A and B. Then*

$$\sum_{C \in F_{A,B}} |C|^2 + d_e(v, C) < \sum_{C \in F} |C|^2 + d_e(v, C).$$

Proof: If $X \in F$ can be reduced to the two cycles A and B then either $d_e(v, A)$ or $d_e(v, B)$ (or both) are equal to $d_e(v, X)$. Without loss of generality let $d_e(v, A) = d_e(v, X)$. But $d_e(v, B) \leq d_e(v, X) + \lfloor \frac{|A|}{2} \rfloor$, since B can be at most half way around A. It follows that

$$\begin{aligned}
|X|^2 + 2(d_e(v, X))|X| &= |A + B|^2 + 2(d_e(v, X))|A + B| \\
&= |A|^2 + |B|^2 + 2|A||B| + 2(d_e(v, X))|A| + 2(d_e(v, X))|B| \\
&= |A|^2 + |B|^2 + 2(d_e(v, A))|A| + 2|B|\,(|A| + d_e(v, X)) \\
&> |A|^2 + |B|^2 + 2(d_e(v, A))|A| + 2|B| \left(\left\lfloor \frac{|A|}{2} \right\rfloor + d_e(v, X) \right) \\
&\geq |A|^2 + 2(d_e(v, A))|A| + |B|^2 + 2(d_e(v, B))|B|.
\end{aligned}$$

Therefore,

$$\sum_{C \in F_{A,B}} |C|^2 + d_e(v, C) < \sum_{C \in F} |C|^2 + d_e(v, C).$$

■

Corollary IV.23 *Let F be a reducible edge-factor of $D = (V, A)$. Then a double Eulerian cycle that defines F does not produce the minimum measure for D.*

Proof: Follows directly from Lemma IV.22 and Theorem IV.19. ■

Equation IV.1 suggests that minimizing the sum of $(|C|^2)$'s in the edge-factor should reduce the measure of a double Eulerian cycle. One would expect the cycles in the edge-factor that yield a double Eulerian walk of minimum measure to manifest the same properties as cycles that minimize the sum of the squares of their lengths.

Lemma IV.24 *Let $\lambda = \{\lambda_1 \geq \lambda_2 \geq \ldots \geq \lambda_t > 0\}$ be a partition of the positive integer K. Then λ is the partition that minimizes $\sum_{i=1}^{t} {\lambda_i}^2$ if and only if* $\lambda_1 - \lambda_t \leq 1$

Proof: Let λ be the partition that minimizes $\sum_{i=1}^{t} {\lambda_i}^2$. Assume $\lambda_1 - \lambda_t \geq 2$. Then

$$\begin{aligned}(\lambda_1 - 1)^2 + (\lambda_t + 1)^2 &= {\lambda_1}^2 + 1 - 2\lambda_1 + {\lambda_t}^2 + 1 + 2\lambda_t \\ &= {\lambda_1}^2 + {\lambda_t}^2 + 2 + 2(\lambda_t - \lambda_1).\end{aligned}$$

Since $\lambda_t - \lambda_1 \leq -2$ it follows directly that $(\lambda_1)^2 + (\lambda_t)^2 > (\lambda_1 - 1)^2 + (\lambda_t + 1)^2$. Therefore, λ_1 and λ_t are not in the partition λ that minimizes the sum of the squares. ■

Loosely speaking, a factor whose cycles are as nearly as possible equal sized should produce the smallest measures. In the next section we apply this heuristic to the Good - de Bruijn digraph to obtain the values of these digraphs.

D. THE VALUE OF THE DE BRUIJN DIGRAPH

Lemma IV.24 shows that an edge-factor that minimizes the sum of the squares of the cycle lengths in a digraph minimizes the differences between cycle length. Every edge-factor of B_n includes at least one cycle of length at least $n + 1$. For example, the $(n + 1)$-tuple $000\ldots 1$ representing an arc in B_n must belong to a cycle of length greater than or equal to $n + 1$. Additionally, if arc α, represented by the $(n+1)$-tuple of all zeros, $000\ldots 0$, is not a member of the cycle $C = (0)$ of length 1, then it is an arc in a reducible cycle. Corollary IV.23 showed that reducible cycles are never permitted in the factor that generates a walk of minimum measure. Therefore, the cycle (0) will appear in the factor yielding the value of the Good - de Bruijn digraph B_n. It follows that the minimum difference between the lengths of cycles is at least n for any edge-factor of B_n that produces the value. Therefore, if F is an edge-factor of irreducible cycles in B_n there have to exist cycles in F whose lengths differ by at least n. We would want the lengths of the cycles in F to be equal, but we now see that F must have a cycle of length 1 and a cycle of at least $n + 1$. The following theorem by Golomb [Ref. 6] insures that the cycle sizes in the edge-factor PCR_{n+1} differ by at most n.

Theorem IV.25 (Golomb) *Let the edge-factor of B_n be generated by the PCR_{n+1}. Then the only cycles appearing are those of length d, where $d|(n+1)$.*

In [Ref. 6] it is shown that there are two possible cycles of length 1, (0) and (1), only one cycle of length 2, (10), two cycles of length 3, (011) and (001), and in general $(\frac{1}{n})\sum_{d|n}\mu(d)2^{\frac{n}{d}}$ cycles of length n, where μ is the Möbius μ-function and the summation is extended over ali divisors of n.

The value of B_n is a function of the edge-factor of B_n and also of the distances to the respective cycles of the factor in the underlying graph of B_n. Further, the distance between cycles in an edge-factor of B_n and the initial vertex v of the minimum double Eulerian cycle affect the value of B_n. A closer examination of Equation IV.1 suggests that if the distance to cycles of larger length in F from the initial vertex v tend to be smaller than the distances to cycles of smaller length, the edge-factor F should generate a smaller measure.

The directed distance, $d_D(x,y)$, between $\mathbf{x} = (x_0, x_1, \ldots, x_{n-1})$ and $\mathbf{y} = (y_0, y_1, \ldots, y_{n-1})$ in B_n is $n - \gamma$ where γ is the longest string of consecutive bits where $(x_{n-\gamma}, x_{n-\gamma+1}, \ldots, x_{n-1}) = (y_0, y_1, \ldots, y_{\gamma-1})$. For example, in B_4, $d_D(0001, 1001) = 4 - 1 = 3$ since the longest string shared by the last consecutive bits in 0001 and the first consecutive bits in 1001 is the single bit, 1. It follows immediately that the directed distance between any two vertices in B_n is at most n.

The distance $d(x,y)$ between $\mathbf{x}$ and $\mathbf{y}$ in B_n is $n - \zeta$ where $\zeta = \max(\gamma, \kappa, \epsilon)$ and

- κ is the longest string of consecutive bits satisfying the equality $(y_{n-\kappa}, y_{n-\kappa+1}, \ldots, y_{n-1}) = (x_0, x_1, \ldots, x_{\kappa-1})$.
- $\epsilon = 2(n - \nu)$ where ν is the longest string of consecutive bits shared by both $\mathbf{x}$ and $\mathbf{y}$.
- γ is defined above.

In B_4, for example, $d(0001, 1001) = 4 - \max(1,0,2) = 2$. This observation allows for a simple way to find the edge-distance $d_e(v, C)$ in G_{B_n} between a vertex v and a cycle C generated from the PCR$_{n+1}$.

Lemma IV.26 *Let F be the PCR_{n+1} edge-factor of B_n. Let δ be the distance from a vertex $v \in V$ to a cycle C in F. Then there exists a directed path between v and $x \in (C)_v$ of length δ in B_n.*

Proof: Follows directly from the discussion above. ■

Lemma IV.27 *Let F be the edge-factor of B_n generated by PCR_{n+1}, $v = \underbrace{00\ldots0}_{\lfloor\frac{n}{2}\rfloor}\underbrace{11\ldots1}_{\lceil\frac{n}{2}\rceil}$ and $\sigma = |C|$ for some $C \in F$. Then $d(v, C) \geq n - \sigma$.*

Proof: If C is a short cycle in PCR$_{n+1}$, i.e., $|C| = \sigma < (n+1)$ and $\sigma|(n+1)$, then at most σ bits of any vertex in $(C)_v$ coincide with any cyclic shift of the n-tuple, $v = \underbrace{00\ldots0}_{\lfloor\frac{n}{2}\rfloor}\underbrace{11\ldots1}_{\lceil\frac{n}{2}\rceil}$. Therefore, the shortest path from v to C is of length at least $n - \sigma$. ■

From Theorem IV.25 we see that the PCR$_{n+1}$ edge-factor possesses the properties that should minimize the sum of the squares of the cycle lengths. From Lemma IV.27, we find that in the PCR$_{n+1}$ edge-factor the lower bound on the distance between larger length cycles and the initial vertex $v = \underbrace{00\ldots0}_{\lfloor\frac{n}{2}\rfloor}\underbrace{11\ldots1}_{\lceil\frac{n}{2}\rceil}$ is always smaller than the lower bound on the distance between smaller cycles and v.

The following two theorems insure that the PCR$_{n+1}$ cycles in B_n are irreducible. Proofs of the theorems can be found in Mykkeltveit [Ref. 20].

Theorem IV.28 *The maximum number of adjacencies between two arbitrary cycles in the PCR$_{n+1}$ is 2.*

Theorem IV.29 *There cannot be more than one adjacency between a cycle with submaximal length* $(< n+1)$ *and any other cycle in the PCR$_{n+1}$. Two cycles, both of which have submaximal length, can not have any adjacencies.*

The ideas and concepts developed in this section and an exhaustive search of all double Eulerian walks in B_n, $1 \leq n \leq 5$ support the following conjecture:

Conjecture IV.30 *Let the PCR$_{n+1}$ cycles be the edge-factor F of the digraph B_n and $v = \underbrace{00\ldots0}_{\lfloor \frac{n}{2} \rfloor}\underbrace{11\ldots1}_{\lceil \frac{n}{2} \rceil}$ the initial vertex in a double Eulerian walk produced by Algorithm D. Then W_v generates the minimum measure over all double Eulerian walks on B_n.*

The PCR$_{n+1}$ factor can be constructed in an efficient fashion using the θ Algorithm of Fredricksen and Kessler [Ref. 26]. Table VI provides the measure of the double Eulerian cycle W_{2_v} generated by Algorithm D using the edge-factor of the PCR$_{n+1}$ and $v = \underbrace{00\ldots0}_{\lfloor \frac{n}{2} \rfloor}\underbrace{11\ldots1}_{\lceil \frac{n}{2} \rceil}$ for $1 \leq n \leq 11$. The numbers for $1 \leq n \leq 4$ were found independently by Harper [Ref. 27].

n	value
1	8
2	24
3	72
4	200
5	524
6	1,400
7	3,420
8	8,352
9	19,476
10	45,232
11	101,722

Table VI. *Conjectured values for* B_n, $1 \leq n \leq 11$

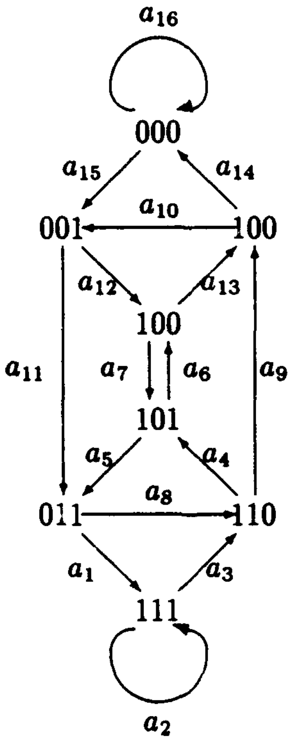


Figure 17. *The Good - de Bruijn digraph* B_3 *with a cycle generated by Algorithm D*

It turns out that the double Eulerian cycles generated by Algorithm D using the edge-factor F of the PCR$_{n+1}$ and $v = \underbrace{00\ldots0}_{\lfloor\frac{n}{2}\rfloor}\underbrace{11\ldots1}_{\lceil\frac{n}{2}\rceil}$ are not the only walks of minimum measure in B_n. In Figure 17, for Example, a double Eulerian walk

$$\begin{aligned}W_1 &= (a_1, a_2, a_2, a_3, a_4, a_5, a_1, a_3, a_4, a_6, a_7, a_6, a_7, a_5, a_8, a_9,\\ &\quad a_{10}, a_{11}, a_8, a_9, a_{10}, a_{12}, a_{13}, a_{14}, a_{15}, a_{12}, a_{13}, a_{14}, a_{16}, a_{16}, a_{15}, a_{11})\end{aligned}$$

is generated by Algorithm D using the edge-factor of PCR$_4$ and $v = 001$. By an exhaustive search, we know this walk has the minimum measure of 72 for any double Eulerian cycle on B_3. W_1 was generated using cycles of lengths 4,4,4,2,1, and 1. The double Eulerian cycle

$$\begin{aligned}W_2 &= (a_1, a_2, a_2, a_3, a_4, a_5, a_1, a_3, a_4, a_6, a_7, a_6, a_7, a_5, a_8, a_9,\\ &\quad a_{14}, a_{15}, a_{11}, a_8, a_9, a_{14}, a_{16}, a_{16}, a_{15}, a_{12}, a_{13}, a_{10}, a_{12}, a_{13}, a_{10}, a_{11})\end{aligned}$$

also has a measure of 72 and uses cycles of lengths 5,4,3,2,1, and 1. Thus, we see that walks of minimum measure are not restricted to the PCR_{n+1} edge-factor. We have not, however, found an edge-factor in B_n where the length of the largest cycle is greater than $n + 2$ that also generates a walk of minimum measure in B_n.

V. A RANDOMNESS PROPERTY OF DE BRUIJN CYCLES

"Living backwards," Alice repeated in great astonishment. "I never heard of such a thing."

"...but there's one great advantage in it, that one's memory works both ways."

"I'm sure mine only works one way," Alice remarked. "I can't remember things before they happen."

"It's a poor sort of memory that only works backwards," the Queen remarked.

Through the looking glass, Chapter 5

A. INTRODUCTION

A strong statistical dependence exits between the predecessor and successor of each state of a binary feedback shift register (FSR). The consequences of randomly selecting each successor state from a choice of two possible states proves have interesting consequences. Selecting the successor state actually can modeled a rather simple Markov process, since the two possibilities can be assigned probabilities. We show that the distribution of runs in a binary de Bruijn cycle coincides with the expected distribution of runs for a binary sequence of length 2^n generated by *randomly* selecting the successor state in a binary FSR.

B. THE BALANCE PROPERTY

A binary sequence S is said to have the *balance property* if the number of 1's in S equals the number of 0's in S [Ref. 6].

Theorem V.1 *A de Bruijn cycle has the balance property.*

Proof: A de Bruijn cycle of length 2^n has the property that every n consecutive bits are different on a given period. Expressed in decimal notation, each n-tuple can be thought of as representing an integer from 0 to $2^n - 1$. In this range there are 2^{n-1} odd integers and 2^{n-1} even integers. Thus, a de Bruijn cycle contains 2^{n-1} 1's and 2^{n-1} 0's and will therefore always possess the balance property. ■

By a simple counting argument we find that the size of the set of all balanced binary sequences of length 2^n is $\binom{2^n}{2^{n-1}}$, where $\binom{n}{r} = C(n,r)$ is the binomial coefficient of n things taken r at a time. A de Bruijn cycle can be normalized by requiring that the sequence begin with exactly n consecutive 0's. There are then $\binom{2^n-n-2}{2^{n-1}-n}$ *normalized* sequences of length 2^n that possess the balance property. This follows since the n consecutive 0's must be preceded and followed by a 1.

For example, when searching the 2^5-long binary sequences for the 2,048 de Bruijn cycles, there are $\binom{2^5}{2^4} = 60,108,390$ sequences of length 2^5 that possess the balance property. There are, however, only $\binom{2^5-7}{2^4-5} = 4,457,400$ normalized balanced sequences of length 2^5.

C. THE RUN PROPERTY

A binary sequence S is said to have the *run property* if among the runs of ZEROs (and ONEs) in S, one-half of the runs are of length one, one-fourth are of length two, one-eighth are of length three, and so on, as long as these fractions provide an integer value for the number of runs.

Theorem V.2 *A de Bruijn cycle has the run property.*

Proof: The run structure of a de Bruijn cycle S_n can be completely determined. A run of ZEROs of length n occurs exactly once in S_n. This run of ZEROs of length n must be preceded and followed by a 1, or the n-tuple $(00\cdots00)$ would appear at least twice in S_n. The n-tuple $(100\ldots0)$ of a 1 followed by $(n-1)$ 0's occurs exactly once in the sequence. This, however, is already accounted for by the run of ZEROs of length n. Thus, there is no run of ZEROs of length $n-1$. To find the number of runs of ZEROs of length k, for $1 \leq k \leq n-2$, we consider all n consecutive terms of the sequence that begin with a 1, then the k-tuple $(00\ldots0)$, and then a 1. Each such run can be made to correspond to an n-tuple t of the form

$$t = 1\underbrace{00\ldots0}_{k}1\underbrace{xx\ldots x}_{n-k-2},$$

where the x's are chosen as arbitrary bits. Since we are free to choose each of the remaining $n-k-2$ components, there are 2^{n-k-2} runs of ZEROs of length

n	Run Distribution: D_n
1	$\{1\}$
2	$\{2\}$
3	$\{1,3\}$
4	$\{1,1,2,4\}$
5	$\{1,1,1,1,2.2,3,5\}$
6	$\{1,1,1,1,1,1,1,1,2,2,2,2,3,3,4,6\}$

Table VII. *Distribution of runs in* S_n

k for $1 \leq k \leq n-2$. With the single run of ZEROs of length n, the result follows. The same distribution holds for runs of ONEs. ■

The multiset of the lengths of the runs of ZEROs (ONEs) in a 2^n-long de Bruijn sequence is denoted by D_n. Table VII is a list of the ZERO and ONE run distributions, D_n, for a sequence $S_n \in \mathcal{S}_n$, where $1 \leq n \leq 6$. Any sequence of length 2^n whose run distribution is a permutation of the multiset D_n has the run property. We alternatively indicate such multisets by specifying the number of times each different type of element occurs. Thus, D_5 can also be denoted by $\{4 \cdot 1, 2 \cdot 2, 1 \cdot 3, 1 \cdot 5\}$ where 4, 2, 1, and 1 are the *repetition numbers* of the entries 1,2,3, and 5, respectively. The repetition number of each element $k \in D_n$ is denoted by $r_{D_n}(k)$, where

$$r_{D_n}(k) = \begin{cases} 2^{n-k-2} & \text{if } 1 \leq k \leq n-2 \\ \\ 1 & \text{if } k = n. \end{cases} \tag{V.1}$$

A binary sequence can be interpreted as a sequence of integers representing the lengths of the alternating runs of ZERO's and runs of ONE's in the sequence. We have seen that a binary 2^n-long sequence S corresponds to the run sequence R [See Chapter III, Section C]. The run sequence R consists of the subsequence $\mathcal{Z}_n$ of the lengths of the runs of ZERO's interleaved with the subsequence $\mathcal{O}_n$ of the lengths of the runs of ONE's. The sequences $\mathcal{Z}_n$ and $\mathcal{O}_n$ for any de Bruijn cycle S_n of length 2^n will always be permutations of D_n because of the run properties of S_n.

In an arbitrary binary sequence of length 2^n possessing the run property, there are $\binom{2^{n-2}}{2^0,2^1,2^2,\ldots,2^{n-3}} = \frac{2^{n-2}!}{2^0!2^1!2^2!\ldots2^{n-3}!}$ ways that the runs of ZEROs may be arranged. We can, however, normalize the binary sequence (i.e., let it begin with the string of n consecutive ZEROs), and then there are only $\binom{2^{n-2}-1}{2^1,2^2,\ldots,2^{n-3}}$ ways to position the 2^{n-2} runs of ZEROs. Hence, there are

$$\binom{2^{n-2}}{2^0,2^1,2^2,\ldots,2^{n-3}}\binom{2^{n-2}-1}{2^1,2^2,\ldots,2^{n-3}}$$

normalized binary sequences of length 2^n possessing the run property where the first factor counts the runs of ONEs and the second factor counts the runs of ZEROs.

As an example, in a normalized binary sequence of length 2^5 possessing the run property, there are $\binom{7}{2,4} = 105$ ways that the runs of ZEROs may be arranged. The runs of ONEs can be independently arranged in $\binom{8}{1,2,4} = 840$ ways. Hence, the 2,048 de Bruijn cycles of length 2^5 are a subset of the $105 \times 840 = 88,200$ normalized binary sequences possessing the run property.

Clearly, not every permutation of $\mathcal{Z}_n$ and $\mathcal{O}_n$ generates a de Bruijn cycle. Interestingly, there is no de Bruijn cycle of length 2^5 with either $\mathcal{Z}_5$ or $\mathcal{O}_5$ equal to (51312121), (51121132), (51123211), (51212131), (51213121), (52113112), or (52311211). This begs the question of why specific permutations of D_n cannot occur as $\mathcal{Z}_n$ or $\mathcal{O}_n$ in a de Bruijn cycle that is answered in the following section. (Note:

When S_n is viewed as a cycle, care must be taken not to break the run sequences across a run when the sequence is cycled.)

D. A RANDOMNESS PROPERTY

A binary sequence of length 2^n having the balance and run property is not necessarily a de Bruijn cycle. For example, the balanced sequence 0000111100110101 has the run property, but clearly this is not a de Bruijn cycle. We show that if the run sequences $\mathcal{Z}_n$ and $\mathcal{O}_n$ of a balance binary sequence with the run property are arranged *randomly*, the resulting binary sequence is de Bruijn. We define what we mean by randomly in the sequel.

The simplest possible non-trivial experiment is one that may result in either of two possibilities. Such an experiment is called a *Bernoulli trial* and the two outcomes are labeled as 1 or 0 (success or failure). This framework is used in what follows.

Let A and B be copies of the multiset D_n. Suppose the elements from the multiset A are viewed as being placed randomly between the positions of another circular permutation of the second multiset B. The interleaving of A and B generates a run sequence R. (We view A as the multiset of the lengths of the runs of ZEROs and B as the multiset of the lengths of runs of ONEs.) Given an arbitrary element j in B, let X_1 be a binary indicator variable defined on j satisfying $X_1 = 1$ if and only if an element 1 from A is to the immediate right (TTIR) of j, otherwise $X_1 = 0$. The *probability function* for the discrete random variable X_1 is

$$p_{X_1}(x) = P(X_1 = x), \quad \text{for} \quad x \in \{0,1\}.$$

Since the element $1 \in A$ accounts for 2^{n-2} of the 2^{n-3} elements in D_n, it follows immediately that $p_{X_1}(1) = .5 = p_{X_1}(0)$. The expected value for X_1 is

$$E[X_1] = \sum_{x \in \{0,1\}} x p_{X_1}(x) = 0(.5) + 1(.5) = (.5).$$

The expected value of X_1 is the probability that a $1 \in A$ is TTIR of an element in B. We know that the expected value of a sum is the sum of the expected values. The expected number of 1's of A that are TTIR of *all* of the entries j in B is denoted $E[1;j] = \frac{1}{2} r_{D_n}(j)$, where $r_{D_n}(j)$ is the repetition number of the element j in the multiset D_n. From Equation V.1, it follows that

$$E[1;j] = \begin{cases} 2^{n-j-3} & \text{if } 1 \leq j \leq n-2 \\ 2^{-1} & \text{if } j = n. \end{cases} \tag{V.2}$$

$E[1;j]$ is an integer except when j is either $(n-2)$ or n. We find it necessary to specify the expected number of 1's that are TTIR of $(n-2)$ or TTIR of (n) in B, denoted $E[1; n-2, n]$, so that an integer value is realized. We then find $E[1; n-2, n] = 1$.

Similarly, we define X_2 on $j \in B$ to be a binary indicator variable where $X_2 = 1$ if and only if a 2 in the multiset A is TTIR of j, otherwise $X_2 = 0$. It follows immediately that $p_{X_2}(1) = .25$, $p_{X_2}(0) = .75$ and $E[X_2] = 0.25$.

The expected number of 2's that are TTIR of all the j's in B, denoted $E[2;j]$ is equal to $(.25) r_{D_n}(j)$. From Equation V.1, it follows that

$$E[2;j] = \begin{cases} 2^{n-j-4} & \text{if } 1 \leq j \leq n-2 \\ 2^{-2} & \text{if } j = n. \end{cases} \tag{V.3}$$

In general, we define the random variable X_k on $j \in B$ to be a binary indicator variable where $X_k = 1$ if and only if a k is TTIR of j, otherwise $X_k = 0$. The probability function for X_k is given as

$$P_{X_k}(1) = \begin{cases} 2^{-k} & \text{if } 1 \leq k \leq n-2 \\ 2^{-k+2} & \text{if } k = n, \end{cases} \tag{V.4}$$

and $P_{X_k}(0) = 1 - P_{X_k(1)}$. The expected value of X_k is given by

$$E[X_k] = \begin{cases} 2^{-k} & \text{if } 1 \leq k \leq n-2 \\ 2^{-k+2} & \text{if } k = n. \end{cases} \tag{V.5}$$

The following Lemma provides the expected number of elements TTIR of each type of element when *random* permutations of D_n are interleaved.

Lemma V.3 *If elements from $A = D_n$ and $B = D_n$ are placed randomly in alternate order, TTIR of all the j's in B we expect to find exactly $E[k,j]$ k's in A, with*

$$E[k,j] = \begin{cases} 2^{n-j-k-2} & \textit{if } 1 \leq j, k \leq n-2 \\ 2^{-k} & \textit{if } 1 \leq k \leq n-2 \textit{ and } j = n \\ 2^{-j} & \textit{if } k = n \textit{ and } 1 \leq j \leq n-2 \\ 2^{-k+2} & \textit{if } k, j = n \end{cases} \tag{V.6}$$

and where the multiset D_n represents the run distribution for a 2^n-long binary sequence possessing the run property.

Proof: Follows directly from the Equations V.1 and V.5 ■

We define the multiset

$$R_1 = \{E[1;1] \cdot 1, E[2;1] \cdot 2, \ldots, E[n-3;1] \cdot n-3\},$$

where R_1 is the submultiset of integers of A that represent the elements that are expected to be TTIR of all the 1's in B. Recall that $E[k;j] = E[X_k]r_{D_n}(j)$. Therefore,

$$R_1 = \{E[X_1]r_{D_n}(1) \cdot 1, E[X_2]r_{D_n}(1) \cdot 2, \ldots, E[X_{n-3}]r_{D_n}(1) \cdot n-3\}.$$

In like manner, $R_2 = \{E[1;2] \cdot 1, E[2;2] \cdot 2, \ldots, E[n-4;2] \cdot n-4\}$.

We define the multiset $R_k = \{E[1;k]\cdot 1, E[2;k]\cdot 2, \ldots, E[n-k-2;k]\cdot k\}$, where $1 \leq k \leq n-2$. The multiset R_k is the submultiset of integers of A that represents the elements that are expected to be TTIR of all the k's in B.

In general, let α be a submultiset of B and define R_α to be the multiset of integers that represents the elements in A that are expected to be TTIR of all the elements in α. If α and β are multisets in B, $R_\alpha = R_\beta$ if and only if $|\alpha|$ and $|\beta|$.

In a similar manner, we denote the multiset of integers that represent the expected elements from A that are TTIR of all the j_1's, j_2's, ..., j_q's, in B by $R_{(j_1,j_2,\ldots,j_q)}$. We define $R_{\overline{k}} = R_{(k+1,k+2,\ldots,n-2,n)}$.

Lemma V.4 *$R_{\overline{k}} = R_k$ for every $1 \leq k \leq n-2$*

Proof: We need to show that $|\overline{k}| = |k|$. By Equation V.1, $r_{D_n}(k) = 2^{n-k-2}$ for $1 \leq k \leq n-2$ and $r_{D_n}(k+1) + \cdots + r_{D_n}(n-2) + r_{D_n}(n) = (\sum_{k=1}^{n-k-3} 2^k) + 1$. Let $\sum\limits_{k=1}^{n-k-3} 2^k = S$. Then $2S - S = S = 2^{n-k-2} - 1$. It follows immediately that $r_{D_n}(k) = r_{D_n}(k+1) + \cdots + r_{D_n}(n-2) + r_{D_n}(n)$. Therefore, $R_{\overline{k}} = R_k$. ■

As an example, let E_6 be a binary sequence whose run sequence consists of alternating terms from two respective multisets D_6 that have been randomly interleaved. Let $\mathcal{Z}_6$ be the sequence of run lengths of ZEROs and $\mathcal{O}_6$ the sequence of run lengths of ONEs. We would expect the following:

1. Exactly four 1's, two 2's, and one 3 in $\mathcal{Z}_6$ are TTIR of the 1's in $\mathcal{O}_6$. $R_1 = \{1,1,1,1,2,2,3\}$.
2. Exactly two 1's and one 2 in $\mathcal{Z}_6$ are TTIR of the 2's in $\mathcal{O}_6$. $R_2 = \{1,1,2\}$.
3. Exactly one 1 in $\mathcal{Z}_6$ is TTIR of the 3's in $\mathcal{O}_6$. $R_3 = \{1\}$.
4. Exactly one 1 in A is TTIR of either the 4 or 6 in $\mathcal{O}_6$. $R_{(4,6)} = \{1\}$.

Similarly, we may continue to determine the expected values of the elements that must be TTIR of the elements in the multisets R_1 and R_2. For example, we define the multiset

$$R_{1/1} = \{E[X_1]E[1;1]\cdot 1, E[X_1]E[2;1]\cdot 2, \ldots, E[X_1]E[n-4;1]\cdot n-4\},$$

where $R_{1/1}$ is the multiset of integers that represents the elements expected TTIR of all the 1's in R_1.

In general we define

$$R_{x_1/\ldots/x_k} = \Big\{(E[X_1])^{k-1}\, E[1;X_1]\cdot 1, (E[X_2])^{k-1}\, E[2;X_1]\cdot 2, \ldots,$$
$$(E[X_p])^{k-1}\, E[p;X_1]\cdot p, \Big\},$$

where $p = n - x_1 - x_2 \cdots - x_k - 2$. The range of values for p insures integer values for the representative numbers in $R_{x_1/\ldots/x_k}$. It follows from Equation V.6 that

$$R_{x_m/\ldots/x_1} = \{1\} \text{ when } x_1 + \cdots + x_m = n - 3. \qquad \text{(V.7)}$$

From Equation V.6, one sees immediately that $E[k,j] = E[j,k]$. Therefore, since multiplication is commutative,

$$R_{x_1/\dots/x_k} = R_{y_1/\dots/y_k}$$

where $y_1, \dots y_k$ is any permutation of $x_1, \dots x_k$. Hence,

$$R_{x_1/\dots/x_k} = R_{(x_1+\dots+x_k)}. \tag{V.8}$$

Continuing the above example, we also expect to find the following in E_6:

1. Exactly two 1's and one 2 in $\mathcal{O}_6$ are TTIR of the 1's in R_1. $R_{1/1} = \{1,1,2\}$.
2. Exactly one 1 in $\mathcal{O}_6$ is TTIR of the 2's in R_1. $R_{1/2} = \{1\}$.
3. Exactly one 1 in $\mathcal{O}_6$ is TTIR of the 1's in R_2. $R_{2/1} = \{1\}$.

This process can continues until $R_{x_1/\dots/x_m} = \{1\}$. This will occur when $x_1 + \dots + x_m = n - 3$.

Concluding the example, we also expect *exactly* one 1 in $\mathcal{O}_6$ TTIR of the two 1's in $R_{1/1}$, therefore, $R_{1/1/1} = \{1\}$. Furthermore, since $R_1 = \{1,1,1,1,2,2,3\}$, it follows that $R_{\overline{1}} = R_{(2,3,4,6)}$ is also equal to $\{1,1,1,1,2,2,3\}$. In a like manner, $R_{\overline{2}} = R_{(3,4,6)} = \{1,1,2\}$, etc.

A binary sequence whose run structure consists of alternating terms from D_n with the expected distribution of elements to the immediate right and left of each like of element in D_n, is said to have the *Expected Value Property*.

A binary sequence of length 2^n with the Expected Value Property exhibits the following:

$$R_1 = \{\underbrace{1,\ldots,1}_{2^{n-4}},\underbrace{2,\ldots,2}_{2^{n-5}},\underbrace{3,\ldots,3}_{2^{n-6}},\ldots\underbrace{(n-4)}_{2^1},(n-3)\},$$

$$R_{\bar{1}} = R_1,$$

$$R_2 = \{\underbrace{1,\ldots,1}_{2^{n-5}},\underbrace{2,\ldots,2}_{2^{n-6}},\ldots\underbrace{(n-5)}_{2^1},(n-4)\},$$

$$R_{\bar{2}} = R_2,$$

$$\vdots$$

$$R_k = \{\underbrace{1,\ldots,1}_{2^{n-(k+3)}},\underbrace{2,\ldots,2}_{2^{n-(k+4)}},\ldots\underbrace{(n-(k+3))}_{2},(n-(k+2)\},$$

$$R_{\bar{k}} = R_k$$

$$\vdots$$

$$R_{n-3} = \{1\}, \text{ and}$$

$$R_{\overline{n-3}} = R_{(n-2,n)} = R_{n-3}.$$

Equations V.7 and V.8 have far-reaching implications. In particular, they provide a connection between the Expected Value Property and de Bruijn Sequences.

Theorem V.5 *A binary sequence S of length 2^n is a de Bruijn cycle if and only if the run sequences of S possess the* Expected Value Property.

Proof: Let S be a binary sequence with the Expected Value Property. Since S consists of alternating terms from two copies of D_n it follows immediately that S has length 2^n.

Let S^j be the sequence formed by cyclically shifting S by j bits to the left. The sequence S has 2^n distinct n-tuples if and only if for every j, $1 \leq j < 2^n$, the sequence formed by the sum $S^j \oplus S$ does not contain a string of n consecutive 0's.

Let S possess the interleaved run sequences $\mathcal{O}_n$ and $\mathcal{Z}_n$, each of which is a separate permutation of D_n, for $1 \leq j < 2^n$. Let T be the longest consecutive string of 0's from the sum sequence $S^j \oplus S$. The length of T is $|T| = t$. Let $R = (r_1, r_2, \ldots, r_{t-1}, r_t)$ and $W = (w_1, w_2, \ldots, w_{t-1}, w_t)$ be the consecutive bits from S^j and S, respectively, such that $R \oplus W = T$.

Since T is the longest consecutive string of zeros, the bit r_0 that immediately precedes R and the bit w_0 that immediately precedes W must differ. Without loss of generality let $r_0 = r_1 = 1$. Define k by $w_1 = w_2 = \cdots = w_{k-1} = w_k$ but $w_k \neq w_{k+1}$. Then k is the length of a run of ONEs in W.

We now show by contradiction that the sequences S has no repeated n-tuples. Since the run lengths in S are elements in D_n, $k \leq n-2$, we know that $R_{(k,k+1,\ldots,n-2,n)} = R_{k-1}$. With $k-1 \leq n-3$, the properties on page 92 and equation V.8 can be used. Let $x_1, x_2, \ldots, x_m$ be the lengths of the runs that coincide after the bits r_k and w_k. Let x_p, where $p < m$ be the length of the last runs in R and W that concide, where $(k-1)+x_1+\cdots+x_n \leq (n-3)$ and $(k-1)+x_1+\cdots+x_p+x_{p+1} > (n-3)$. Let $(k-1)+x_1+\cdots+x_p = n-d$, where

$d > 3$. Since $R_{k-1/x_1/\ldots/x_p} = R_{n-d}$, it follows that $x_{p+1} > d-3$. The length of the longest run that coincides between R and W TTIR of R_{n-d}, however, is $d-3$. We find, therefore, that $(k-1)+x_1+\cdots+x_m = n-g \leq n-3$. The length of the longest string of bits that coincides TTIR of R_{n-g}, however, is $g-2$. Since $(k-1)+x_1+\cdots+x_m = n-g$, it follows that $k+x_1+\cdots+x_m+g-2 = n-1$ is the length of the longest string of bits that coincide. Therefore, S has no repeated n-tuples and is therefore a de Bruijn cycle.

Conversely, let S be a de Bruijn cycle. Since S is de Bruijn, it consists of alternating terms from D_n and S has the Run Property. Consider the runs of ONEs of length r where $0 < r \leq n-2$. Each such run can be made to correspond to an n-tuple along the sequence S of the form

$$0\underbrace{11\cdots1}_{r}0\underbrace{xx\cdots x}_{n-r-2},$$

where the x's are arbitrary bits. The number of runs of ZEROs of length j where $n-r-j \geq 0$ that are TTIR of the runs of ONEs of length r is $2^{n-r-j-2}$. This coi:.:ides with the expected integer values given by Lemma V.6. Hence, the Expected Value Property for S follows immediately from the Run Property and the uniqueness of the n-tuples in a de Bruijn cycle. Therefore, a binary sequence of length 2^n with the expected value property is a de Bruijn cycle. ■

Interleaving any permutation of the run sequences of D_n for $1 \leq n \leq 3$ generates a de Bruijn cycle. For $n = 4$ we find that a sequence with the expected value property must satisfy the following condition with respect to its run sequences: if the sequence of runs of ONEs/ZEROs has 2 consecutive 1's in it then the sequence of runs of ZEROs/ONEs cannot have 2 consecutive 1's. This condition provides a very simple way to generate the 16 de Bruijn cycles of length 2^n. There are 3 different normalized permutations of D_4: (4,2,1,1), (4,1,1,2), and (4,1,2,1). There are 8 permutations of D_4 with consecutive 1's : (4,2,1,1), (4,1,1,2), (2,4,1,1), (2,1,1,4), (1,4,2,1), (1,2,4,1), (1,1,4,2), and (1,1,2,4). Finally, there are 4 permutations of D_4 with no consecutive 1's: (4,1,2,1), (2,1,4,1), (1,4,1,2), and (1,2,1,4). The 16 de Bruijn cycles are generated by:

1. Interleaving the two normalized runs sequences from D_4 having consecutive 1's with the four runs sequences from D_4 not having consecutive 1's, and
2. Interleaving the single normalized run sequence from D_4 having no consecutive 1's with the eight runs sequences from D_4 having consecutive 1's.

Table VIII displays these sequences.

In the general case, although the expected value property can be used to generate de Bruijn cycles, it does not provide a more efficient means to construct them than appears elsewhere.

4 2 1 1	4,1,2,1	1111000011010010
	2,1,4,1	1111001101000010
	1,4,1,2	1111011000010100
	1,2,1,4	1111011001010000
4,1,1,2	4,1,2,1	1111000010100110
	2,1,4,1	1111001010000110
	1,4,1,2	1111010000101100
	1,2,1,4	1111010010110000
4,1,2,1	4,2,1,1	1111000010011010
	4,1,1,2	1111000010110100
	2,4,1,1	1111001000011010
	2,1,1,4	1111001010110000
	1,4,2,1	1111010000110010
	1,2,4,1	1111010011000010
	1,1,4,2	1111010110000100
	1,1,2,4	1111010110010000

Table VIII. *The de Bruijn cycles of length 2^4 generated by run sequences*

VI. A MEMORYLESS ALGORITHM TO GENERATE DE BRUIJN SEQUENCES

"To change and change for the better are two different things."

German Proverb

A. INTRODUCTION

In this chapter we present an algorithm to construct a subset of all the binary de Bruijn sequences of length 2^{n+1}. A comprehensive survey of previous work on this subject can be found in [Ref. 28]. A common approach to this process is to join cycles together to form a full cycle. This same general approach will be followed here. The algorithm presented is not limited to joining pure cycles, however, using the PCR feedback function and its pure cycles produces the most efficient algorithm. Unlike the *Universal Algorithm* [Ref. 29], the representative vertex for each cycle is determined by a distance function on the digraph B_n, rather than by finding the *smallest* element among all the vertices on a cycle. Our algorithm produces an Eulerian cycle in B_n. As this is isomorphic to a Hamiltonian cycle in B_{n+1}, our algorithm allows for an interplay between vertices and arcs that provides a systematic way to join cycles. As a consequence, we decompose the Good - de Bruijn digraph into an edge-factor to insure each arc is traversed, rather than into a factor that partitions the set of vertices.

B. BACKGROUND FOR SIMILAR APPROACHES

Golomb [Ref. 6] describes the key-sequence method to generate de Bruijn sequences. One generates the set of span n de Bruijn sequences, S_n, from preference functions (P_1^*, P_2^*) that are obtained from feedback formulas and from the preference functions (P_1, P_2) for the span $n - 1$ sequences. The key-sequence method requires considerable memory storage to generate the sequences because every de Bruijn sequence of smaller span is utilized in order to generate the de Bruijn sequences of span n.

Fredricksen [Ref. 30] shows how to generate 2^{2n-5} full cycles of length 2^n from the *prefer ones* sequence using the PCR_n. The method uses $6n$ bits of storage, and n units of time to produce the next bit from a given n bits.

Etzion and Lempel [Ref. 31] show how to generate $2^{(k)g(n,k)}$ full cycles of length 2^n from the PCR_n using $3n + (k)g(n,k)$ bits of storage, where k is a constant in the range $1 \leq k \leq 2^{\frac{n-4}{2}}$, $g(n,k) \approx (n-2\log \mathrm{k})(1-\frac{1}{1+\log \mathrm{k}})$. The time required to produce the next bit from the last n bits is $O(n)$.

C. JOINING OF CYCLES

Common to all of the above techniques is the notion of cycle joining. When the two possible predecessors of a state both map into the same state by the feedback function f we say f is *singular*. Any feedback function that is not singular is therefore *nonsingular*. The following are equivalent conditions:

1. f is nonsingular.
2. f produces no branches.
3. f is a one to one mapping.
4. f is an onto mapping.
5. Every state has a unique predecessor and a unique successor.
6. f produces only cycles.

The following theorem [Ref. 6] is stated without proof:

Theorem VI.1 (Golomb) *A necessary and sufficient condition for distinct states to have distinct successors is that the corresponding (nonsingular) feedback function $f\colon B^{n+1} \rightarrow \{0,1\}$ can be written*

$$f(z_0, z_1, \ldots, z_n) = z_0 \oplus f_1(z_1, z_2, \ldots, z_n), \tag{VI.1}$$

for some function $f_1\colon B^n \rightarrow \{0,1\}$

Any edge-factor of the digraph B_n represents a feedback function that satisfies equation (VI.1). Furthermore, any feedback function satisfying (VI.1) will have distinct successors and the function therefore yields an edge-factor in the corresponding Good - de Bruijn digraph.

Since B_n is a 2-regular strongly connected digraph, when B_n is decomposed into an edge-factor F containing two or more cycles, each distinct cycle is adjacent (i.e., arc disjoint and sharing a common vertex) to some other cycle. By the following lemma, it is always possible to join together two adjacent cycles in a edge-factor to form a single cycle.

Lemma VI.2 *Let F be an edge-factor of the digraph D. Let (R) and (T) be distinct adjacent cycles in F with $x \in (R)_v$ and $x \in (T)_v$. The cycles (R) and (T) are joined into a single cycle when the respective successors of the vertex x in each cycle are interchanged.*

Proof: Let the following sequence of vertices, $(R) = (r_1, r_2, \ldots, r_l)$ and $(T) = (t_1, t_2, \ldots, t_m)$, represent adjacent cycles. Since the cycles are adjacent there is a vertex x that lies in each of $(R)_v$ and $(T)_v$. Let $r_i = t_j = x$. The sequence of vertices $(r_1, r_2, \ldots, r_i, t_{j+1}, t_{j+2}, \ldots, t_j, r_{i+1}, \ldots, r_l)$ created by exchanging the successors of x, i.e., exchanging outgoing arcs of x, forms a single cycle. ■

To illustrate Lemma VI.2 consider the PCR$_3$ where $f(z_0, z_1, z_2) = z_0$ on B_2 shown in Figure 18. This function generates the edge-factor of Figure 19A. The cycles, represented by a sequence of arcs, are:

1. $C_1 = (0)$
2. $C_2 = (001)$
3. $C_3 = (011)$
4. $C_4 = (1)$

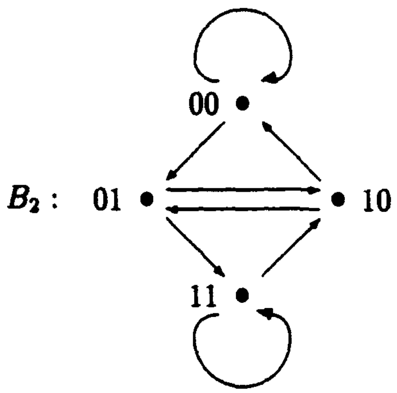


Figure 18. *The Good - de Bruijn digraph B_2 as one edge-factor*

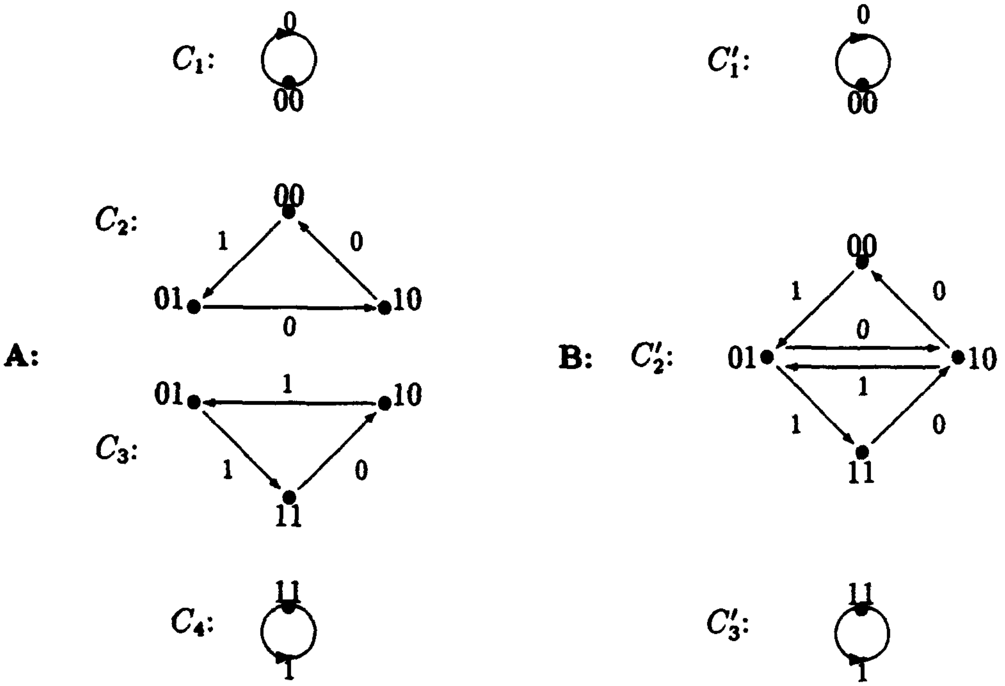


Figure 19. *Edge-factors*

The cycles C_2 and C_3 contain the common vertices 01 and 10. By Lemma VI.2 a single cycle is formed when the successors, 10 and 11, of the vertex 01 are interchanged between the cycles C_2 and C_3, respectively. The cycles now, represented by a sequence of arcs, are:

1. $C_1' = (0)$
2. $C_2 = (101001)$
3. $C_3' = (1)$

This new edge-factor is shown in Figure 19B. Therefore, the problem of joining cycles from an edge-factor reduces to finding the vertices common between adjacent cycles.

To construct the algorithm that generates a de Bruijn sequence, a representative vertex for each cycle plays an important role and must be identified. The *cycle representative* for each cycle is any vertex in the cycle that has the smallest directed distance from a designated vertex v_D in B_n. (The designated vertex v_D is an arbitrary but fixed vertex in B_n.) The cycle representative $(C_i)_R$ of the cycle (C_i) is defined to be any vertex $v_i \in (C_i)_v$ such that for the given designated vertex $v_D \in V$,

$$d_D(v_D, v_i) = \min_{v \in (C_i)_v} d_D(v_D, v).$$

If there is more than one vertex on a cycle of minimum directed distance from v_D, we can define a unique representative by selecting, say, the vertex of largest/smallest decimal value. (Recall that the directed distance $d_D(x, y)$ between $\mathbf{x} = (x_0, x_1, \ldots, x_{n-1})$

and $\mathbf{y} = (y_0, y_1, \ldots, y_{n-1})$ in B_n is $n - \gamma$, where γ is the longest string of consecutive bits where $(x_{n-\gamma}, x_{n-\gamma+1}, \ldots, x_{n-1}) = (y_0, y_1, \ldots, y_{\gamma-1})$.)

Lemma VI.3 shows that cycle representatives are unique to a cycle unless the representative is the designated vertex v_D.

Lemma VI.3 *For an edge-factor $F = \{C_1, C_2, \ldots, C_m\}$ of $B_n = (V, A)$, let $C_i \in F$ where $d_D(v_D, (C_i)_v) > 0$ and v_D is the designated vertex. Then there exists a cycle $C_j \in F$ adjacent to C_i such that $d_D(v_D, (C_j)_v) < d_D(v_D, (C_i)_v)$.*

Proof: The proof is analogous to the proof of Lemma IV.17. ■

Corollary VI.4 *Every vertex that is the representative of its cycle but not equal to the designated vertex v_D is incident with exactly two cycles in any edge-factor.*

Proof: This follows immediately from Lemma VI.3. ■

Lemma VI.3 can be applied with distance functions other than the directed distance from a designated vertex. The representative vertex $(C_i)_R$ of the cycle C_i can be defined as either the vertex for which $d_D((C_i)_R, v_D,) = \min_{v \in (C_i)_v} d_D((C_i)_R, v_D)$ or the vertex for which $d(v_D, (C_i)_R) = \min_{v \in (C_i)_v} d(v_D, (C_i)_R)$. In either of these cases the properties of the representative vertex remain the same and the concepts in the next section can be applied.

D. THE CYCLE JOINING ALGORITHM

Consider a feedback shift register with $n+1$ stages. The ith state of the FSR_{n+1} is denoted by $S_i = (s_i, s_{i+1}, \ldots, s_{i+n})$. Given a nonsingular feedback function f on the register, the next state following $S_i = (s_i, s_{i+1}, \ldots, s_{i+n})$ is obtained from the Cycle Joining Algorithm as follows:

Cycle Joining Algorithm

Input: A feedback function f for a FSR_{n+1}, the current state $S_i = (s_i, s_{i+1}, \ldots, s_{i+n})$ of the register, and a list of representative vertices.

Output: The next state S_{i+1}.

```
(C1)    if $(s_{i+1}, \ldots, s_{i+n})$ designates a representative vertex,
            then $S_{i+1} := f(S_i) \oplus 1$
            else $S_{i+1} = f(S_i)$.
```

(This cycle joining algorithm is analogous to others, however, it was originally used by Fredricksen [Ref. 32].)

We can apply Lemmas VI.2 and VI.3 to provide a simple way to construct a de Bruijn cycle of length 2^{n+1}. At each shift of the FSR_{n+1} we check the last n bits in the register. The n bits represent a vertex in B_n. If the vertex is a cycle representative and this is the first occurrence, we expand the current cycle, by joining an adjacent cycle in the edge-factor to the current cycle. If the vertex is not a cycle representative we continue along the current cycle. If the vertex is a cycle representative and this is the second occurrence, we *close* the current cycle and enter a previous cycle. The algorithm constructs an Eulerian walk W_{v_D} along B_n by successively selecting cycles from an edge-factor to be added and traversed. This is accomplished by stepping along a nonsingular shift register until the current state identifies a cycle representative at which time the Cycle Joining Algorithm is executed and a new cycle is added.

We now describe an algorithm (Algorithm H) that constructs a Eulerian walk along B_n beginning at the designated vertex v_D. Algorithm H adjoins cycles from an edge-factor F to create the Eulerian walk. This algorithm is similar to other common algorithms [Ref. 29] that join cycles from a factor and hence create a Hamiltonian cycle along B_n. Here the edge-factor method allows us to use the properties of *distance* in the digraph to uniquely identify a common vertex on adjacent cycles in B_n. Loosely, speaking, Algorithm H moves *forward*, entering and adding cycles at a cycle representative of increasingly greater distance from the vertex v_D for as long as this is possible. Each cycle representative is on exactly two cycles. Algorithm H enters a cycle representative $(D)_R$ for the first time along an arc on a cycle C where $d_D(v_D, (C)_v) < d_D(v_D, (D)_v)$. The algorithm immediately departs the vertex $(D)_R$ along an arc on the cycle D. When the cycle representative $(D)_R$ is entered for the second time, it is along an arc on the cycle D and the vertex $(D)_R$ is departed along an arc on the cycle C. When it is no longer possible to enter a new cycle at a greater distance, the algorithm traverses each arc on the last adjoined cycle and then backtracks to the previous cycle at its cycle representative. The backtracking continues to "close" previously entered cycles until the first cycle from which it is possible to again go forward to enter and add new cycles. A cycle C could conceivably be entered at several places along the walk. Algorithm H, however, does not allow a cycle C to be entered *for the first time* at any vertex other than at the cycle representative $(C)_R$.

ALGORITHM H [Constructs an Eulerian walk along B_n]

Input: A Good - de Bruijn digraph $B_n = (V, A)$, a designated vertex $v_D \in V$ where $v_D = (v_1, v_2, \ldots, v_n) \in B^n$, and a shift register with a nonsingular feedback function f of degree $n+1$.

Output: A sequence $(0, s_1, s_2, \ldots, s_{2^{n+1}-1})$ constituting an Eulerian cycle through B_n.

Algorithm H

Initially set the register to $S_1 = (0, v_1, v_2, \ldots, v_n) = (s_1, s_2, \ldots, s_n)$. Step the register once to produce $S_2 = (s_1, s_2, \ldots, s_n, f(S_1 \oplus 1))$, and set $i = 3$.

H1 $S_{i-1} = (s_{i-1}, s_i, \ldots, s_{i+n})$ lies on cycle C. We examine each of the states on the cycle until a state is found that identifies a vertex of greater distance from the designated vertex than the current state. If no such state exists (i.e. the current state identifies the cycle representative), then go to [H1a], else go to [H1b].

H1a $S_i = (s_{i-1}, s_i, \ldots, s_{i+n-1}, f(S_{i-1}))$; go to [H2].

H1b $S_i = (s_{i-1}, s_i, \ldots, s_{i+n-1}, f(S_{i-1}) \oplus 1)$; go to [H2].

H2 Increment i.

H2a If $i < 2^{n+1}$ go to [H1]; else end.

We now show that the walk constructed by Algorithm H visits every edge in B_n exactly once.

Corollary VI.5 *Let $F = \{C_1, C_2, \ldots, C_m\}$ be an edge-factor of $B_n = (V, A)$ and v_D is the designated vertex. Furthermore, let v_D be a vertex in exactly one cycle in F. We define a graph T_{B_n} as follows: the vertices in T_{B_n} are the cycle representatives of the cycles in F, and there is an edge between vertices x and y in T_{B_n} if and only if $x, y \in (C_i)_v$ for some cycle $C_i \in F$. Then T is a tree.*

Proof: The order of T_{B_n} is m, the number of cycles in F. By Theorem VI.4 for each cycle $C_i \in F$ where $d_D(v_D, (C_i)_v) > 0$ there exists a unique cycle $C_j \in F$ adjacent to C_i such that $d_D(v_D, (C_j)_v) < d_D(v, (C_i)_v C)$. Therefore, T_{B_n} is connected and the size of the edge set of T_{B_n} is $m - 1$. Hence, T_{B_n} is a tree. In like manner, in the case where v_D is in exactly 2 cycles in F, T_{B_n} is also a tree. ■

Thus, given an edge-factor F of the digraph $B_n = (V, A)$ and a vertex $v_D \in V$, Algorithm H joins every cycle in F and thereby constructs an Eulerian walk along B_n beginning at the vertex v_D. (Algorithm H is in fact a depth first search of T_{B_n}.) Since T_{B_n} is connected, every arc is traversed exactly once.

To illustrate the algorithm, consider a FSR_5 and the pure cycling register defined by the feedback function $f(z_0, z_1, z_2, z_3, z_4) = z_0$. Let the designated vertex v_D in B_4 be 0011. The edge-factor contains the eight cycles and their respective rep esentatives as shown in Table IX. (Note: the algorithm does not compute the table of representatives *a priori*, rather, representatives are computed on line.) The cycle (00101) has two vertices 1001 and 1010 each of the same minimum directed

Cycle	Representative
(0)	0000
(00001)	1000
(00011)	0011
(00111)	0011
(01111)	0111
(01011)	0110
(00101)	1001
(1)	1111

Table IX. *Cycles and representatives for a PCR_5, where v_D* = 0011

distance from v_D. We arbitrarily select the vertex 1010 to be the representative, as it is of largest decimal value. Applying the Cycle Joining Algorithm to this FSR_5, the cycles are joined together to form one cycle, shown in Figure 20. The de Bruijn sequence representing this cycle is

00011111011100110100101011000001.

Changing the cycle representative of (00101) to 1001, the cycles are joined to form a different sequence

00011111011100101001101011000001.

Continuing the example with the same feedback function, when the designated vertex is changed to 1001 and using the smallest representative 0101 for the cycle (01011) a different sequence is obtained

01001111101110010110101000110000.

Finally, with $v_D = 1001$ and selecting 0110 as the representative of the cycle (01011) we get the sequence

$$01001111101110010100011010110000.$$

Each of these sequences are different (i.e., none is a cyclic shift of another). It is not always the case that distinct v_D's yield different sequences. Clearly, identical sets of representative vertices generate identical sequences. The size of the class of de Bruijn sequences generated by the pure cycling register using Algorithm H has not been determined. Using just the feedback function of PCR_4, however, 12 of the 16 de Bruijn sequences of length 16 can be obtained by changing the vertex v_D.

E. THE FEEDBACK FUNCTIONS

Using the PCR_n, the Cycle Joining Algorithm requires storage of n bits for the current state S_i, n bits for cycling the current state, and n bits to store the current representative on a cycle, for $3n$ bits of storage. The greatest amount of time in this algorithm is spent to determine the representative vertex on each cycle. The worst case occurs when the current state is the cycle representative and the entire cycle must be traversed. Therefore, having an edge-factor with relatively short cycles is desirable. For an arbitrary nonsingular feedback function, the length of the longest cycle can be expected to be very large [Ref. 6]. Jansen et.al. provides an approximate count on the number of linear functions that yield cycles whose maximum cycle length is $4n$ [Ref. 29].

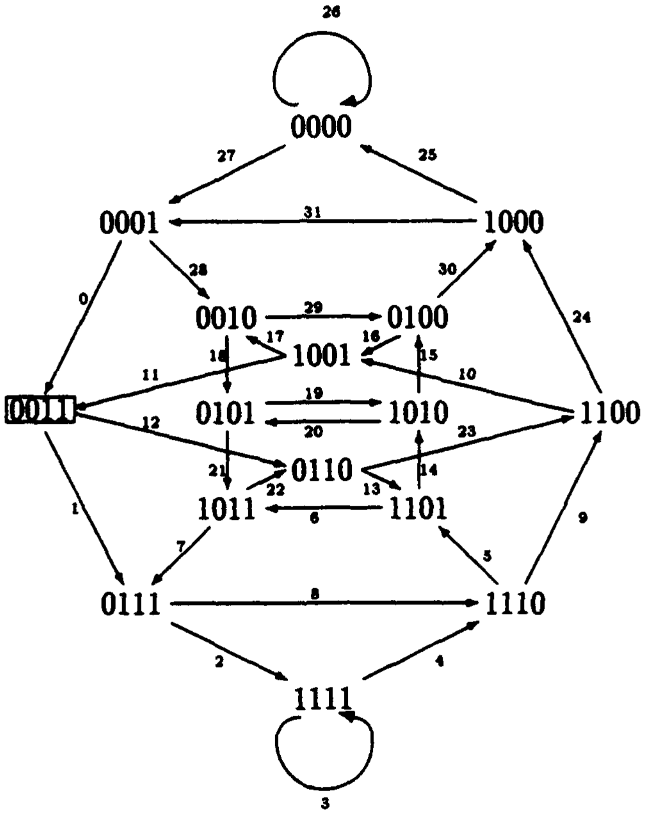


Figure 20. *An Eulerian cycle in B_4*

VII. CONCLUSIONS

"Results! Why, I have got a lot of results. I know several thousand things that won't work."

Thomas Edison

A. SYNOPSIS OF THE THESIS

The binary ie Bruijn cycle provides a special case of extracting distinct n-tuples from a 2^n-long binary sequence. Generalizing the de Bruijn property, i.e., extracting n bits that are not necessarily consecutive along the cycle, was the fundamental idea motivating this work. There appears to be no previously published work analyzing the set of complete cycles, a set that includes the well-known de Bruijn cycles.

In Chapter III, the framework is developed to systematically *comb* from a sequence every n-tuple appearing at successive positions along a subsequence of the sequence. We found that the run structure for all but one class of complete sequences is completely determined. The set of complete sequences with $m < n$ consecutive teeth defines a cycle visiting every arc an equal number of times along an appropriate Good - de Bruijn digraph. Only the case for $m = n - 1$ is thoroughly discussed in Chapter III. Chapter III concludes with a probabilistic argument that shows that the size of the set of complete sequences for a specific (l, n)-comb is substantially smaller than the size of the set of 2^n-long de Bruijn cycles. The class of de Bruijn sequences is the largest class of complete sequences. This surprising result is not intuitively obvious, given the randomness properties of the eligible sequences.

Chapter IV provides a comprehensive survey on double Eulerian cycles along graphs and digraph. The necessary and sufficient conditions for a graph or digraph to have a double Eulerian cycle are given. We describe a class of double Eulerian cycles along a graph (digraph) that minimizes the difference of visitation times along each edge (arc). The necessary and sufficient conditions are found to generate a *minimum* double Eulerian cycle. A closed formula using the edge-status of a graph is shown to provide the value of a graph. An algorithm is presented that, given the *appropriate* edge-factor, generates the minimum double Eulerian cycle along a digraph. A closed formula is provided for the value of a Eulerian digraph. A conjecture for the appropriate edge-factor to generate the value of the Good - de Bruijn digraph is made.

Chapter V provides the necessary and sufficient conditions for the permutation of the run sequences in a de Bruijn cycle. We see that the structure of distinct n-tuples is a randomness property of the runs in the sequence. This structure is a specific case of the property that each arc in a walk defined by a class of complete cycles is equally likely to be taken along an appropriate digraph B_n.

A new efficient algorithm for the generation of classical de Bruijn sequences is developed in Chapter VI. The algorithm uses only $3n$ bits of storage to generate a de Bruijn cycle of length 2^n. Although this algorithm adjoins cycles in the usual way, the class of sequences generated is different from that found by previous algorithms.

B. OPEN QUESTIONS

The scope of potential research in the area of generalized de Bruijn cycles is vast. The very fundamental question of the existence of nonclassical de Bruijn cycles for each value of n remains to be answered. A general formula describing the size of the set of nonclassical sequences continues to be an elusive problem.

The formulation of an algorithm to generate a class of complete cycles for each value of n would be a significant contribution in this area of research.

There are many *different* (l, n)-combs available to extract distinct n-tuples. Further analysis on specific combs would be a valuable extension of the work done in Chapter III.

Proving the conjecture in Chapter IV on the value of the Good - de Bruijn digraphs would complete an important contribution to the analysis of double Eulerian cycles on directed graphs. More generally, finding the *appropriate* edge-factor of a digraph to generate the value of a graph would be a very nice result.

The class of sequences generated by the algorithm in Chapter VI has not been fully examined. The relation, if any, between these sequences and sequences generated by previously algorithms would be an interesting result. Determining the designated vertex, representative vertices, and the distance function to generate the *prefer ones* sequences, for example, could provide insight into a new algorithm.

These are but a few of the promising research areas that arise from the challenge of understanding complete sequences. A thorough understanding of nonclassical de Bruijn sequences is an endeavor to find structure in *randomness*, a statement that might be make of every research endeavor!

This appears to be the first study of complete sequences, and it portends to be a rich area of continuing research.

APPENDIX A. SUMMARY OF SOME PREVIOUS WORK ON CIRCUITS IN DIRECTED GRAPHS

"In all things, there is a law of cycles"

Tacitus

The following is a brief summary of the derivation of Theorem III.3. This theorem allows one to easily determine the number of ways to traverse each arc of an Eulerian digraph exactly σ times. The development follows the work by van Aardenne-Ehrenfest and de Bruijn [Ref. 12].

Let S_m be the permutation group (or symmetric group) on m elements. If S is a subset of S_m then the number of elements of S of order m is denoted by $||S||$ and the total number of elements in S by $|S|$. A subset $\mathcal{D}$ of S_m is called a *$\mathcal{D}$-set* (in S_m), whenever it has the property that $||s\mathcal{D}||$ has the same value for all $s \in S_m$.

Let k and n be natural numbers, and take $m = kn$. Consider the set E_m of m elements, divided into k systems, each containing n objects. The set H denotes the subgroup of S_m consisting of all $k!(n!)^k$ permutations with the property that if $h \in H$ then ha and hb belong to the same system whenever a and b belong to the same system. In other words, H transforms systems into systems.

The following theorems from [Ref. 12] are stated with proof.

Theorem A.1 (van Aardenne-Ehrenfest and de Bruijn) *H is a $\mathcal{D}$-set in S_m.*

Let R denote the set of all permutations in S_m with the property that the n objects of each system are transformed into objects of n different systems. In other words, if $r \in R$ where a and b belong to the same system, then ra and rb belong to different systems. It can also be shown that R is a *$\mathcal{D}$-set* in S_m.

Let $D = (V, A)$ be a regular Eulerian digraph where $\text{in}(x) = \text{out}(x) = \alpha$, $\forall_x \in V$. Two cycles are considered identical whenever the arcs of the first cycle are a cyclic permutation of the arcs of the second.

The number of Eulerian cycles in D is denoted by $||D||$. A permutation P of the set of arcs of $A = (a_1, \ldots, a_q)$ is called *conservative* (with respect to D), if $Pa_i = a_j$ always implies that a_i is adjacent to a_j. Let E_o be an arbitrary but fixed Eulerian cycle in D. The set of all conservative permutations of D can be represented as $\mathcal{B}E_o$, where $\mathcal{B} = \mathcal{B}_1 \times \ldots \times \mathcal{B}_{|V|}$ is the group of all permutations where $\mathcal{B}_i$ is the group of permutations that permute arcs having the same initial vertex v_i, but leave invariant all other arcs. Therefore, any Eulerian cycle determines a conservative permutation and any conservative permutation determines an Eulerian cycle. Hence, $||D|| = ||\mathcal{B}E_o||$. Let $B_1, \ldots, B_{|V|}$ be subsets of $\mathcal{B}_1, \ldots, \mathcal{B}_{|V|}$ respectively, where

$$B = B_1 \times \ldots \times B_{|V|}.$$

The set of Eulerian cycles defined by B, denoted ${}_B||D||$, are the cycles corresponding to a permutation bE_o where $b \in B$ and E_o is a fixed conservative permutation. Using this notation, we have

$$||D||_1 =_B ||D||, \quad {}_B||D|| = ||BE_o||_1.$$

If for each i, B_i is a $\mathcal{D}$-set in $\mathcal{B}_i$, then B is called *normal.* The following result provides a relationship between the set of all conservative permutations and a subset of consvervative permutations.

Theorem A.2 (van Aardenne-Ehrenfest and de Bruijn) *If B is normal, then*

$$\frac{{}_B||D||}{|B|} = \frac{{}_{\mathcal{B}}||D||}{|\mathcal{B}|},$$

where $|B|$ and $|\mathcal{B}|$ denote the number of elements of B and $\mathcal{B}$, respectively.

Recall that $D = (V, A)$ is an Eulerian regular directed graph with $|V|$ vertices and $|A|$ arcs. If $\text{in}(v_i) = \sigma$ for $1 \leq i \leq |V|$, then $|A| = \sigma|V|$.

Let λ be a positive integer. Then by $D_{(\lambda)}$ we denote the graph that arises from D if we replace any arc (x,y) by λ arcs (x,y). We see that $D_{(\lambda)}$ has $|V|$ vertices and $\lambda|A| = \lambda\sigma|V|$ arcs. The set of λ arcs arising from each arc in D are said to form a bundle.

We consider 3 categories of cycles that traverse every arc in $D_{(\lambda)}$. Each of these categories is normal in the sense described above.

1. $B_i = \mathcal{B}_i$, where $\mathcal{B}_i$ is the permutation group of order $((\lambda\sigma)!)$.

2. $B_i = H_i$, where H_i is the subset of $\mathcal{B}_i$ that associates bundles with bundles. That is, if a and b are in the same bundle, then ha and hb are in the same bundle where $h \in H_i$. The order of H_i is $\sigma!(\lambda!)^{\sigma}$.

3. $B_i = R_i$, where R_i is the subset of $\mathcal{B}_i$ that associates the arcs of each bundle with σ different bundles. That is, if a and b are in the same bundle, then ra and rb are in different bundles where $r \in R_i$. The order of R_i is $|R|$.

We have by Theorem A.2

$$\frac{{}_{\mathcal{B}}||D_{(\lambda)}||}{((\lambda\sigma)!)^{|V|}} = \frac{{}_{H}||D_{(\lambda)}||}{(\sigma!)^{|V|}(\lambda!)^{|E|}} = \frac{{}_{R}||D_{(\lambda)}||}{|R|}. \tag{A.1}$$

Recall that ${}_{\mathcal{B}}||D_{(\lambda)}|| = ||D_{(\lambda)}||_1$. The number ${}_{H}||D_{(\lambda)}||$ is proportional to the number $||D||_1$. It can be seen that each Eulerian cycle of D arises from $\lambda^{-1}(\lambda!)^{|V|}$ different cycles in $||D_{(\lambda)}||$. Hence,

$$_{H}||D_{(\lambda)}|| = \lambda^{-1}(\lambda!)^{|V|}||D||_1.$$

By Equation A.1 we have,

$$||D_{(\lambda)}||_1 = \lambda^{-1}||D||_1\left(\frac{(\lambda\sigma)!}{\sigma!}\right)^{|V|}. \tag{A.2}$$

A σ-cycle in D is a cycle containing each edge of D exactly σ times. A σ-cycle is called restricted if it happens that any pair of adjacent edges of D appears just once in the cycle. The number of different σ-cycles can be determined from Equation A.2. A difficulty arises from the fact that a σ-cycle may be periodic of period $|E|d$, where d is a divisor of σ. Let $c(\rho)$ denote the number of ρ-cycles in D with the period $|E|\rho$.

For each ρ-cycle in D there arise $(\rho!)^{|E|}$ Eulerian cycles in $D_{(\rho)}$. It follows immediately that

$$||D_{(\rho)}||_1 = \sum_{d|\rho} \frac{d}{\rho} c(d)(\rho!)^{|E|}. \tag{A.3}$$

From the Möbius inversion formula,

$$c(\rho) = \sum_{d|\rho} \frac{d}{\rho} \mu \left(\frac{\rho}{d}\right) (d!)^{-|E|} ||D_{(d)}||_1, \tag{A.4}$$

and the number of unrestricted ρ-cycles equals

$$\sum_{d|\rho} c(d) = \frac{1}{\rho} \sum_{d|\rho} \phi \left(\frac{\rho}{d}\right) (d!)^{-|E|} d ||D_{(d)}||_1, \tag{A.5}$$

where ϕ is Euler's totient function. Evaluating $||D_{(\rho)}||_1$ using Equation A.2, the number of unrestricted ρ-cycles in a σ-regular directed graph $D = (V, A)$ is given by

$$||D||_\rho = \frac{1}{\rho} \sum_{d|\rho} \phi \left(\frac{\rho}{d}\right) \left(\frac{(\sigma d)!}{(d!)^\sigma \sigma!}\right)^{|V|} ||D||_1$$

where ϕ is the Euler's totient function and the summation is extended over all divisors of ρ.

APPENDIX B. A DIGRAPH ISOMORPHIC TO B_n

A friend is ... a second self.

Cicero

In the Good - de Bruijn digraph B_n each vertex is labeled with a unique binary n-tuple. There is an arc $(\mathbf{x},\mathbf{y})$ from vertex $\mathbf{x} = (x_0, x_1, \ldots, x_{n-1})$ to vertex $\mathbf{y} = (y_0, y_1, \ldots, y_{n-1})$ in B_n if and only if $(x_1, x_2, \ldots, x_{n-1}) = (y_0, y_1, \ldots, y_{n-2})$. Equivalently, we can label each vertex $x = (x_1, x_2, \ldots, x_{n-1})$ with the unique integer i that is the decimal representation of x defined by $i = \sum_{j=0}^{n-1} x_j 2^{n-j-1}$. There is an arc from vertex i to vertex j if and only if $j \in \{2i, 2i+1\} \bmod 2^n$.

We define $\overline{B}_{n-(k+1)}$ for $0 < k+1 < n$, to be a 2-regular directed graph with $2^{n-(k+1)}$ vertices and 2^{n-k} arcs. We label each vertex in $\overline{B}_{n-(k+1)}$ with a unique 2^{k+1} element set

$$\{2^{k+1}i, 2^{k+1}i+1, \ldots, 2^{k+1}i+2^{(k+1)}-1\} \text{ where } 0 \leq i \leq 2^{n-(k+1)}-1.$$

There is an arc from the vertex $\{2^{k+1}i, 2^{k+1}i+1, \ldots, 2^{k+1}i+2^{(k+1)}-1\}$ to the vertex $\{2^{k+1}j, 2^{k+1}j+1, \ldots, 2^{k+1}i+2^{(k+1)}-1\}$ in $\overline{B}_{n-(k+1)}$ if and only if

$$2\nu \bmod 2^n \in \{(2^{k+1}j), (2^{k+1}j+1), \ldots, 2^{k+1}j+2^{(k+1)}-1\},$$

where $\nu \in \{2^{k+1}i, 2^{k+1}i+1, \ldots, 2^{k+1}i+2^{(k+1)}-1\}$.

Let $V(B_n)$ be the set of vertices in B_n where the vertices are represented by the integers i where $0 \leq i \leq 2^n - 1$. Let $V(\overline{B}_{n-(k+1)})$ denote the set of vertices in $\overline{B}_{n-(k+1)}$. Furthermore, let $A(B_n)$ and $A(\overline{B}_n)$ denote the set of arcs in B_n and $\overline{B}_n$, respectively.

Lemma B.1 *The function ϕ: $V(B_m) \rightarrow V(\overline{B}_{n-(k+1)})$ for $m = n - (k+1)$, is defined by the rule*

$$v\phi = \{2^{k+1}v, 2^{k+1}v + 1, \ldots, 2^{k+1}v + 2^{(k+1)} - 1\} \bmod 2^n.$$

Then ϕ: $V(B_m) \rightarrow V(\overline{B}_{n-(k+1)})$ is one to one.

Proof: Let $i, j \in V(B_m)$, with $i \not\neq j$. Then

$$\begin{aligned} i\phi &= \{2^{k+1}i, 2^{k+1}i + 1, \ldots, 2^{k+1}i + 2^{(k+1)} - 1\} \bmod 2^n, \text{ and} \\ j\phi &= \{2^{k+1}j, 2^{k+1}j + 1, \ldots, 2^{k+1}j + 2^{(k+1)} - 1\} \bmod 2^n. \end{aligned}$$

Assume $i\phi = j\phi$. Since $2^n > 2^m$, it follows immediately that $i = j$. Therefore, ϕ is 1-1. ■

Lemma B.2 *For the rule ϕ as defined in Lemma B.1, ϕ: $V(B_n) \rightarrow V(\overline{B}_{n-(k+1)})$ is onto.*

Proof: Let $\{2^{k+1}i, 2^{k+1}i + 1, \ldots, 2^{k+1}i + 2^{(k+1)} - 1\} \bmod 2^n \in V(\overline{B}_n)$. Then $\{2^{k+1}i, 2^{k+1}i + 1, \ldots, 2^{k+1}i + 2^{(k+1)-1}\} \bmod 2^n = i(\phi)$. ■

Lemma B.3 *The function θ: $A(B_n) \to A(\overline{B}_{n-(k+1)})$ for $m = n - (k+1)$, is defined by the rule*

$$\begin{aligned}(i,j)\theta = \quad & (\{\ 2^{k+1}i, 2^{k+1}i+1, \ldots, 2^{k+1}i + 2^{(k+1)} - 1\} \bmod 2^n, \\ & \{\ 2^{k+1}j, 2^{k+1}j+1, \ldots, 2^{k+1}j + 2^{(k+1)} - 1\} \bmod 2^n),\end{aligned}$$

where $(i,j) \in A(B_n)$ if and only if $(i(\phi), j(\phi)) \in A(\overline{B}_{n-(k+1)})$. Then θ: $A(B_n) \to A(\overline{B}_{n-(k+1)})$ is one to one.

Proof: Let $(i,j), (i',j') \in A(B_n)$. with $(i,j) \neq (i',j')$. Then

$$\begin{aligned}(i,j)\theta = \quad & (\{\ 2^{k+1}i, 2^{k+1}i+1, \ldots, 2^{k+1}i + 2^{(k+1)} - 1\} \bmod 2^n, \\ & \{\ 2^{k+1}j, 2^{k+1}j+1, \ldots, 2^{k+1}j + 2^{(k+1)} - 1\} \bmod 2^n)\end{aligned}$$

and

$$\begin{aligned}(i',j')\theta = \quad & (\{\ 2^{k+1}i', 2^{k+1}i'+1, \ldots, 2^{k+1}i' + 2^{(k+1)} - 1\}, \\ & \{\ 2^{k+1}j', 2^{k+1}j'+1, \ldots, 2^{k+1}j' + 2^{(k+1)} - 1\} \bmod 2^n).\end{aligned}$$

Assume $(i,j)\theta = (i',j')\theta$. It follows immediately that $(i,j) = (i',j')$. Therefore, θ is 1-1. ■

Lemma B.4 *For the rule θ as defined in Lemma B.3, θ: $A(B_n) \to A(\overline{B}_{n-(k+1)})$ is onto.*

Proof: Let

$$(\{\ 2^{k+1}i, 2^{k+1}i+1, \ldots, 2^{k+1}i+2^{(k+1)}-1\} \bmod 2^n,$$
$$\{\ 2^{k+1}j, 2^{k+1}j+1, \ldots, 2^{k+1}j+2^{(k+1)}-1\} \bmod 2^n) \in V(\overline{B}_{n-(k+1)}.$$

Then

$$(\{\ 2^{k+1}i, 2^{k+1}i+1, \ldots, 2^{k+1}i+2^{(k+1)}-1\} \bmod 2^n,$$
$$\{\ 2^{k+1}j, 2^{k+1}j+1, \ldots, 2^{k+1}j+2^{(k+1)}-1\} \bmod 2^n) = (i,j)(\theta).$$

■

Theorem B.5 *The digraphs B_m and $\overline{B}_{n-(k+1)}$ for $m = n-(k+1)$, are isomorphic.*

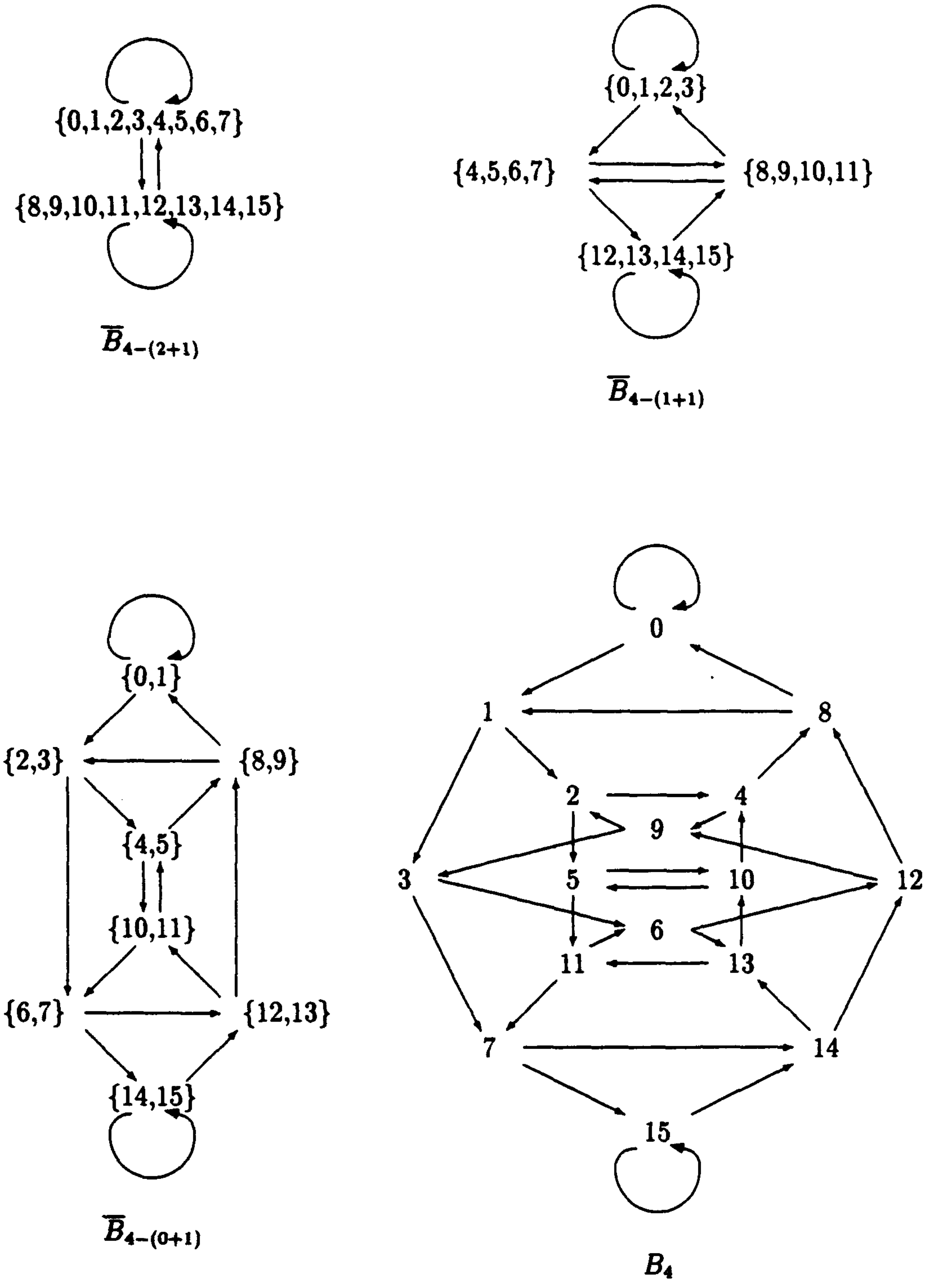
Proof: The function ϕ: $V(B_n) \rightarrow V(\overline{B}_{n-(k+1)})$ defined by the rule

$$v\phi = \{2^{k+1}v, 2^{k+1}v+1, \ldots, 2^{k+1}v+2^{(k+1)}-1\} \bmod 2^n$$

and the function θ: $A(B_n) \rightarrow A(\overline{B}_n)$ defined by the rule

$$(i,j)\theta = \ (\{\ 2^{k+1}i, 2^{k+1}i+1, \ldots, 2^{k+1}i+2^{(k+1)}-1\},$$
$$\{\ 2^{k+1}j, 2^{k+1}j+1, \ldots, 2^{k+1}j+2^{(k+1)}-1\} \bmod 2^n)$$

are bijections. It follows by definition that the digraphs B_n and $\overline{B}_{n-(k+1)}$ are isomorphic. ■



$\overline{B}_{4-(2+1)}$

$\overline{B}_{4-(1+1)}$

$\overline{B}_{4-(0+1)}$

B_4

Figure 21. *Good - de Bruijn digraphs* B_4*,* $\overline{B}_{4-(0+1)}$*,*$\overline{B}_{4-(1+1)}$*, and* $\overline{B}_{4-(2+1)}$

REFERENCES

[1] H. Beker and F. Piper. *Cipher Systems.* Northwood Books, London, 1982.

[2] F.W. Sinden. Sliding Window Codes. *AT&T Bell Laboratories Technical Memorandum: File case 38194-23*, 1985.

[3] S. Golomb (editor). *Digital Communications with Space Applications.* Prentice-Hall, New Jersey, 1964.

[4] S. Lin and D. J. Costello. *Error Control Coding: Fundamentals and Applications.* Prentice-Hall, New Jersey, 1983.

[5] R. C. Dixon. *Spread Spectrum Systems.* John Wiley & Sons, New York, 1976.

[6] Solomon W. Golomb. *Shift Register Sequences.* Aegean Park Press, Laguna Hills, Calif., 1982.

[7] L. R. Ford. A Cyclic Arrangement of m-tuples. *RAND Corporation Report*, P-1071, 1957.

[8] Abraham Lempel. On a Homomorphism of the de Bruijn Graph and Its Applications to the Design of Feedback Shift Registers. *IEEE Trans. on Computers*, C-19(12):1204–1209, 1970.

[9] Harold Fredricksen. Generation of the Ford Sequence of length 2^n. *J. Comb Theory, Ser A*, 12:152–153, 1972.

[10] R. E. Kibler. $2^{n+1} - 2$-Cycles from $(n + 1)$-Registers. *NSA Technical Journal*, R41, 1972.

[11] Neal Koblitz. *A Course in Number Theory and Cryptography.* Springer-Verlag, New York, 1988.

[12] T. van Aardenne-Ehrenfest and N. G. de Bruijn. Circuits and Trees in Oriented Linear Graphs. *Simon Stevin*, 28:203–217, 1951.

[13] N. G. de Bruijn. A Combinatorial Problem. *Nederl. Akad. Wetensch. Proc.*, 49:758–764, 1946.

[14] N.G. de Bruijn. Acknowledgement of Priority to C. Flye Sainte-Marie on the Counting of Circular Arrangements of 2^n Zeros and Ones that show each n-Letter Word exactly once. *Technological University Eindhoven Netherlands, Dept of Mathematics T.H. Report 75-WSK-06*, 1, 1975.

[15] C. Flye Sainte-Marie. Solution to Problem Number 58. *l'Intermédiaire des Mathématiciens*, 1:107–110, 1894.

[16] W. Mantel. Resten van wederkerige reeksen. *Nieuw Arch. Wisk.*, 2:172–184, 1897.

[17] M.H. Martin. A Problem in Arangements. *Bull. Amer. Math. Soc.*, 40:859–864, 1934.

[18] I. J. Good. Normal Recurring Decimals. *J. London Math. Soc*, pages 167–169, 1946.

[19] J. A. Bondy and U. S. R. Murty. *Graph Theory with Applications.* American Elsevier, New York, N.Y., 1976.

[20] J. Mykkeltveit. A Proof of Golomb's Conjecture for the de Bruijn Graph. *J. Combin. Theory*, 13:40–45, 1973.

[21] John E. Hopcroft and Jeffrey D. Ullman. *Introduction to Automata Theory, Languages, and Computation.* Addison-Wesley, Reading, Massachusetts, 1979.

[22] Richard A. Brualdi. *Introductory Combinatorics.* North Holland, New York, N.Y., 1992.

[23] R. E. Tarjan. Depth-First Search and Linear Graph Algorithms. *SIAM J. Comput.*, 1:146–160, 1972.

[24] Fred Buckley and Frank Harary. *Distance in Graphs.* Addison-Wesley, New York, N.Y., 1990.

[25] G. R. T. Hendry. On Graphs with Prescribed Median. *J. Graph Theory*, 9:477–481, 1985.

[26] H. Fredricksen and I. Kessler. An Algorithm for Generating Necklaces of Beads in Two Colors. *Discrete Mathematics*, 61:181–188, 1986.

[27] Lawrence Harper. Personnal communication.

[28] Harold Fredricksen. A Survey of Full Length Nonlinear Shift Register Cycle Algorithms. *SIAM Rev.*, 24:195–221, 1982.

[29] C.J.A. Jansen, W. G. Franx, and D. E. Boekee. An Efficent Algorithm for the Generation of de Bruijn Cycles. *IEEE Trans on Information Theory*, 37:1475–1478, 1991.

[30] Harold Fredricksen. A Class of Nonlinear de Bruijn Cycles. *J. Comb Theory, Ser A*, 19:192–199, 1975.

[31] Tuvi Etzion and Abraham Lempel. Algorithms for the Generation of Full-Length Shift-Register Sequences. *Technical Report 245 - Dept of Computer Science - Israel Institute of Technology*, 1982.

[32] Harold Fredricksen. The Lexicographically Least de Bruijn Cycle. *Journal of Combinatorial Theory*, 9(1):1-5, 1970.

INITIAL DISTRIBUTION LIST

1. Defense Technical Information Center 2
   Cameron Station
   Alexandria, VA 22304-6145

2. Library, Code 52 2
   Naval Postgraduate School
   Monterey, CA 93943-5002

3. Department Chairman, Code MA 1
   Department of Mathematics
   Naval Postgraduate School
   Monterey, CA 93943-5002

4. Prof H. Fredricksen, Code MA/Fs 25
   Naval Postgraduate School
   Monterey, CA 93943-5002

5. Prof R. Franke, Code MA/Fe 1
   Naval Postgraduate School
   Monterey, CA 93943-5002

6. Prof Y. Kanayama, Code CS/Ka 1
   Naval Postgraduate School
   Monterey, CA 93943-5002

7. Prof S. Lawphongpanich, Code OR/Lp 1
   Naval Postgraduate School
   Monterey, CA 93943-5002

8. Prof C.W. Rasmussen, Code MA/Ra 1
   Naval Postgraduate School
   Monterey, CA 93943-5002

9. Dr. Aparna W. Higgins 1
   Dept. of Mathematics
   University of Dayton
   300 College Park
   Dayton, OH 45469-2316

10. LTC G. Krahn 4
U.S.Military Academy
Department of Mathematical Sciences
West Point NY 10996-1786